ARMY RESEARCH LABORATORY

VHDL Modeling of PRC-70 Radio ASICs for Reverse Engineering

Jared M. Brodsky

ARL-TR-419 August 1994

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The citation of trade names and names of manufacturers in this report is not to be construed as official Government endorsement or approval of commercial products or services referenced herein.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1994	Technical Report 1991-1994

4. TITLE AND SUBTITLE
VHDL Modeling of PRC-70 Radio ASICs for Reverse Engineering

5. FUNDING NUMBERS
PE: 62705A
PR: AH94
TA: Ci.11.01

6. AUTHOR(S)
Jared M. Brodsky

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army Research Laboratory (ARL)
Electronics & Power Sources Directorate (EPSD)
ATTN: AMSRL-EP-I
Fort Monmouth, NJ 07703-5601

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-419

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Army Research Laboratory has been researching methods to maintain fielded legacy systems built with technology that is no longer available in the commercial market. One method under examination is using the VHSIC Hardware Description Language (VHDL) to document circuits for future redesign and reengineering for efficient reprocurement of spare electronic components. This report describes a case where structural VHDL modeling methods were studied while documenting the structure of three recently reengineered ASICs. These three ASICs form part of an antenna coupler board which is used in the AN/PRC-70 special operation forces radio. The report describes the steps used in setting up the structural models from the technology information models through the cell library models to the overall structure models of each ASIC. Through the description of the creation of the models a reader will be able to understand how the models allow future reengineering of these ASICs to be technology independent. The report also describes the steps taken to validate the VHDL models and verify them against the actual design drawings using data taken from the results of simulating both. A number of obstacles were revealed during the simulation and verification of the models. An account of those obstacles and the solutions to them are also described in the report.

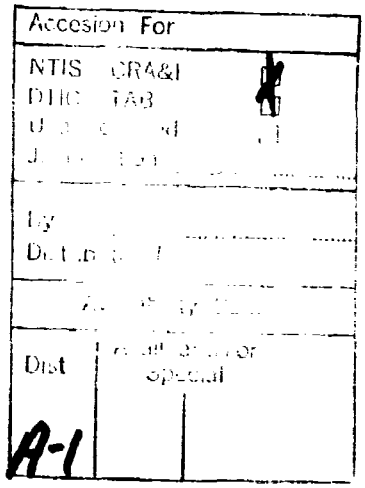
14. SUBJECT TERMS
VHSIC Hardware Description Language; Reverse Engineering; ASIC Simulation, ASIC documentation.

15. NUMBER OF PAGES
166

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

Table of Contents



List of Figures

List of Tables

1.0 Introduction

Recently, the Army has identified a need for researching methods to maintain older electronic equipment built with technology that is no longer commercially available. Consequently, Army Research Laboratory (ARL) has been investigating reverse engineering methods to deal with this problem. One reverse engineering method ARL has been investigating is the use of the VHSIC Hardware Description Language (VHDL) for documentation of circuits for future redesign and reengineering for efficient electronics reprocurement. The VHSIC Hardware Description Language (VHDL) became an Institute of Electrical and Electronics Engineers (IEEE) standard in 1987. Since then, the Department of Defense (DOD) has required that digital application specific integrated circuits (ASICs) be documented by means of structural and behavioral VHDL descriptions. Behavioral VHDL descriptions must describe the input/output behavior at a sufficiently detailed level to permit the behavioral description to be used within a larger VHDL model for test generation and fault grading of the contained model.[1] The VHDL descriptions must also meet the guidelines stated in the Data Item Description (DID) for VHDL deliverables.[2]

One recent vehicle to drive the research of VHDL for reverse engineering documentation was modeling three ASICs recently redesigned by EPSD to emulate three older ASICs that the Army could no longer procure. The U.S. Army Communications-Electronics Command (CECOM) identified a requirement for five hundred of each of the three replacement ASICs. These three ASICs form part of the antenna coupler for the AN/PRC-70 radio, which is used by Army Special Operation Forces. The three ASICs were a relay and pulse counter (SM-B-746131), a time-out and delay counter (SM-B-746395), and a controller circuit (SM-B-746396). The relay and pulse counter consists of two relay counters, two pulse counters, a clock generation circuit for the pulse counters, a sensitivity detection circuit, and a reset and switch required signal generator. The time-out and delay counter consists of a time-out counter, a pulse delay counter, a pulse generator, and a pulse turnoff counter. The controller circuit consists of a program counter and the logic

1. Mil Std 454 Section 4.1.5
2. Document DI-EGDS-0811

necessary to control the associated circuitry (including the other two ASICs) required to couple an antenna to the transmitter of the PRC-70.

The three older ASICs were designed by Riehl Time Corporation, Whippany, NJ in the middle to late 1970s. They were manufactured in 7 micron CMOS and required a power supply voltage of 9 to 11 volts with an absolute maximum rating of 15 volts on every pin (instead of the 5 volts available in most of today's technology). The required test frequency and operating frequency in the radio was 25 kHz. They were also required to meet MIL-STD-883 class B qualifications. In addition to the problem of finding a commercial source to meet the voltage, quality, and quantity requirements at a reasonable cost, the new ASICs also had to match the original timing specifications to ensure that they would operate correctly in the radio over the range of operating conditions. Often, that is a problem with older components. The original circuits had large path delays. Circuit path delays are usually decreased as the technology to fabricated electronic components changes.

Initially, the original specifications for the circuits were needed for the primary redesign. CECOM delivered logic diagrams and test vector tables along with the source control drawings of the original circuits to ARL's Electronics and Power Sources Directorate (EPSD). The diagrams, tables, and drawings served as documentation for reference in redesigning, simulating, fabricating, and testing circuits that could be used as replacements for the original circuits. Tobyhanna Army Depot (TOAD), where the antenna coupler boards were to be rebuilt, provided several sets of the original ASICs for use as reference devices as well as access to the capability to test the redesigned circuits in boards and radios.

ARL used the Defense Logistics Agency (DLA) sponsored Generalized Emulation of Microcircuits (GEM) process, developed by David Sarnoff Research Center (DSRC) in Princeton, NJ, to fabricate new PRC-70 chips. The GEM process implements a gate array with a capability to insert resistive and capacitive (RC network) components in order to control a circuit's path delays. This option allows a gate by gate and cell by cell timing match to the original circuit elements in order to increase the capability of the process to efficiently emulate original ASICs in form, fit and function. DSRC customized the GEM

process slightly to include a power supply voltage requirement of approximately 10 volts to complete the emulation of the original PRC-70 circuits.

In order to document the design in a consistently readable form, all three ASICs were modeled in structural VHDL to match the schematic. The purpose of the VHDL modeling was also to develop a general approach to designing and validating any digital circuit. The VHDL models serve as a basis for future design and validation of PRC-70 ASICs. Each VHDL model was validated by comparing its simulation results with: (1) the results of its respective ASIC's schematic simulation and (2) the ASIC's physical test results from the IC tester. The VHDL models and documentation to help explain them must also satisfy the requirements of the VHDL Data Item Description (DID) before they are delivered. The deliverables include all levels of the model's hierarchy including macro cells and leaf level modules. These VHDL modules were created for the PRC-70 project and for future work with the GEM cell family. This report discusses a method for modeling and simulating designs in VHDL for documenting redesigned circuits originally produced from obsolete technology. It documents an approach to organizing a model and its hierarchy. It also gives an account of problems that can occur and their solutions. The discussion will use the PRC-70 models as an illustration of the methods.

2.0 Modeling Approach

The methodology for structural modeling of integrated circuits is outlined in Figure 1.

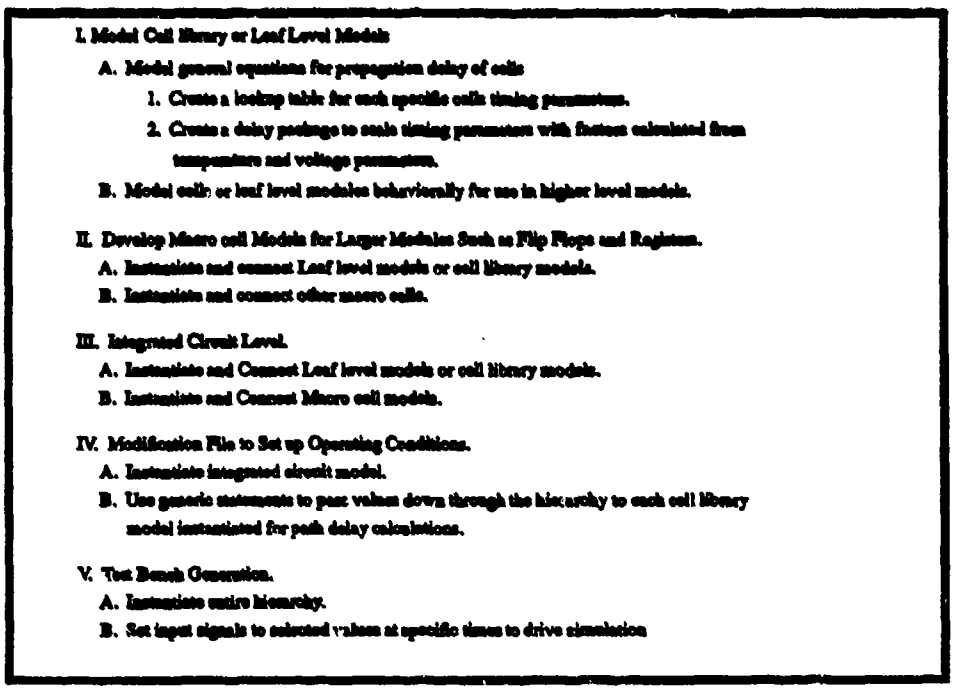


FIGURE 1. Outline of Structural Modeling Method

To illustrate the methods described in Figure 1, a description of the application of the method for modeling the PRC-70 follows.

Creation of the PRC-70 models began with modeling gates and other primitive cells in the cell library. The library contained all the cells available in the GEM process. First, in order to modularize the problem, reduce the amount of necessary information for the modeling to deal with at any given time, and simplify the modeling effort, timing calculations were separated from the structural models. Three packages were created that contain functions, constants, and data to compute timing and delay values for each GEM library cell. The three packages were **gem_constants**, **gem_delays**, and **tables** (Appendix A). Figure 2 illustrates how the packages were used to perform the necessary calculations for each cell. The source control drawings from CECOM and the GEM cell library specifications from DSRC contained all the parameters and equations necessary to derive the arithmetic functions in the packages. These functions calculated delays due to ambient temperature, fanout, voltage variation, and intrinsic delays due to rise and fall time. If any parameter values were out of a specific range declared in the package, the

package would produce a message to notify the user and then use the minimum or maximum value of the parameter for its calculations depending on which end of the range the parameter value exceeded.

The first package, gem_constants, contained a load constant, C_L, used in the delay calculations. The gem_delays package performed the necessary calculations for gate timing and delays. The tables package contained tables for each library cell. Each table consisted of slope M and Y intercept values Y_θ of the characteristic loading curves of the respective cell. The gem_delays package used these values along with the value of C_L from gem_constants and values for fanout F to compute a delay due to loading t_l with the equation:

$$t_l = FMC_L Y_\theta \qquad \text{(EQ 1)}$$

Then the load delay is scaled by a rise or fall delay scaling factor due to the power supply voltage ($S_{r(VDD)}$ and $S_{f(VDD)}$ respectively), V_{DD} and also a rise or fall delay scaling factor caused by temperature ($S_{r(temp)}$) and $S_{f(temp)}$), T to give the total rise or fall time ($t_{r(total)}$) and ($t_{f(total)}$) for the respective cell:

$$t_{r(total)} = t_l S_{r(VDD)} S_{r(temp)} \qquad \text{(EQ 2)}$$

and

$$t_{r(total)} = t_l S_{r(VDD)} S_{r(temp)} \qquad \text{(EQ 3)}$$

where the equations for $S_{r(VDD)}$ and $S_{f(VDD)}$ are:

$$S_{r(VDD)} = 0.64798 + 0.070739 + 1.0/V_{DD} + 8.4494/V_{DD}^2 \qquad \text{(EQ 4)}$$

and

$$S_{f(VDD)} = 1.9983 - 8.6695/V_{DD} + 18.394/V_{DD}^2 \qquad \text{(EQ 5)}$$

and the equations for rise and fall delay scaling factors due to temperature T in °K are:

$$S_{r(temp)} = 0.52246 + (1.23x10^{-3})T + (1.25x10^{-6})T^2 \quad \text{(EQ 6)}$$

and

$$S_{f(temp)} = 0.50673 + (1.6967x10^{-3})T + (1.3889x10^{-7})T^2 \quad \text{(EQ 7)}$$

The **gem_delays** package will return the rise delay and fall delay values scaled in nanoseconds which will control the timing of the model during simulation. F, V_{DD} and temperature T were obtained from generic parameters in each GEM cell behavioral description.

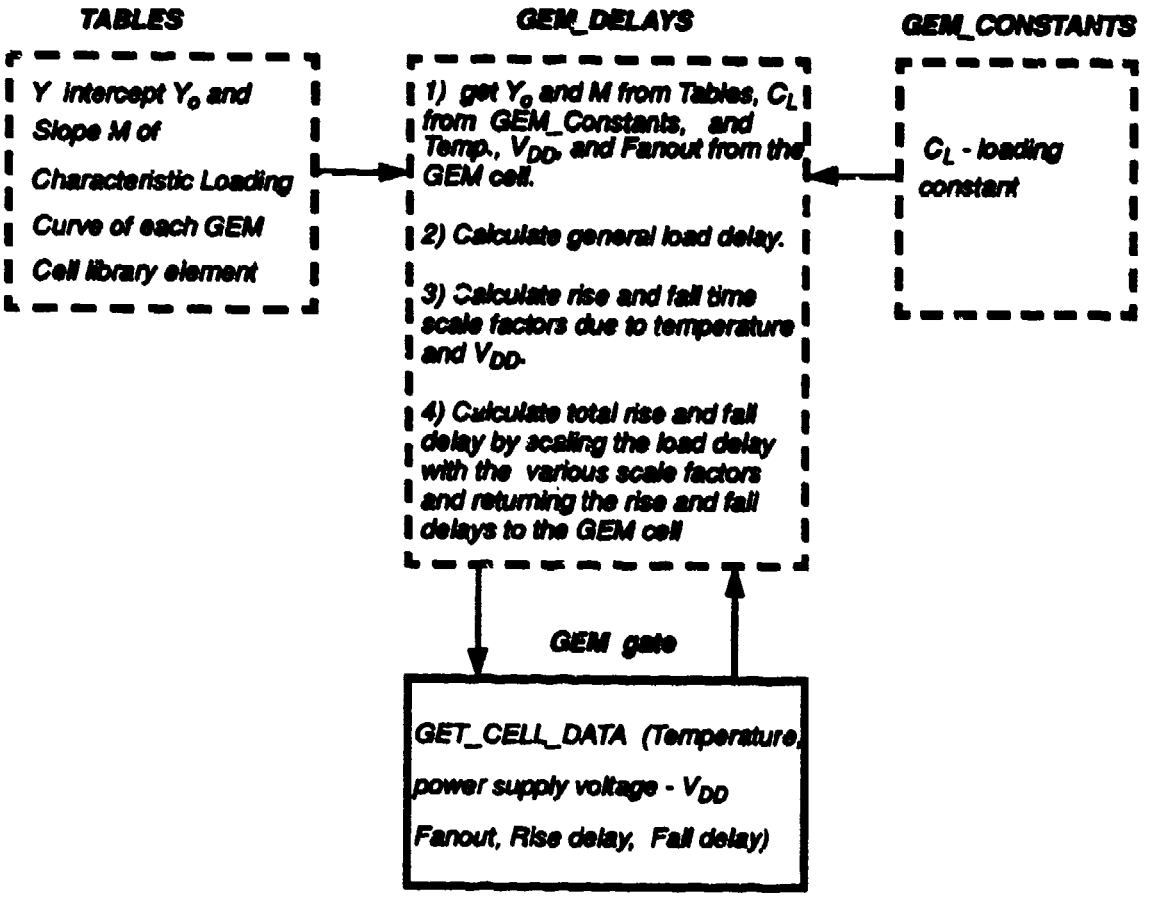


FIGURE 2. Technology Packages Modeling Construction

Following the package development, each element of the GEM cell library was modeled with behavioral VHDL code (Appendix A). The models contained basic behavior and timing that was calculated by calling the equation functions in the **gem_delays** package. For example, here in Figure 3 is the code for the behavior of the GEM inverter library cell. The code contains variables INV_RISE_DELAY and INV_FALL_DELAY to calculate timing of the cell by calling functions from package **gem_delays** through GET_CELL_DATA.

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
use STD.Standard.all;
entity INV is
  generic (DELAY:  time; FANOUT: integer; VDD real; TEMPERATURE:
        real; INV_TABLE: cell_values);
  port (INPUT: in  bit:= '0'; OUTPUT: out bit:= '1');
end INV;

architecture BEHAVIORAL of INV is
    signal cell_version: String(1 to 16):="          ";
    signal rise_del,fall_del:  real:= 0.0;
    signal INV_RISE_DELAY,INV_FALL_DELAY:  time:= 1 fs;

begin
    GET_CELL_DATA(INV_TABLE,DELAY,LOAD,VDD,TEMPERATURE,FANOUT,
                  cell_version, rise_del,fall_del,
                  INV_RISE_DELAY,INV_FALL_DELAY);

    process
    begin

      if (INPUT = '0') then
        OUTPUT <= not INPUT after INV_RISE_DELAY;
      else
        OUTPUT <= not INPUT after INV_FALL_DELAY;
      end if;
      wait on INPUT;
    end process;
end BEHAVIORAL;
```

FIGURE 3. Behavioral VHDL Code for Inverter Cell

The second step involved the creation of macro cells (Appendix A) using some of the GEM cell library elements. Figures 4 through 9 show the schematics of macro cells. Table 1 contains a list of the macro cells and the numbers of each individual GEM cell library element or macro cell instantiated in each macro cell.

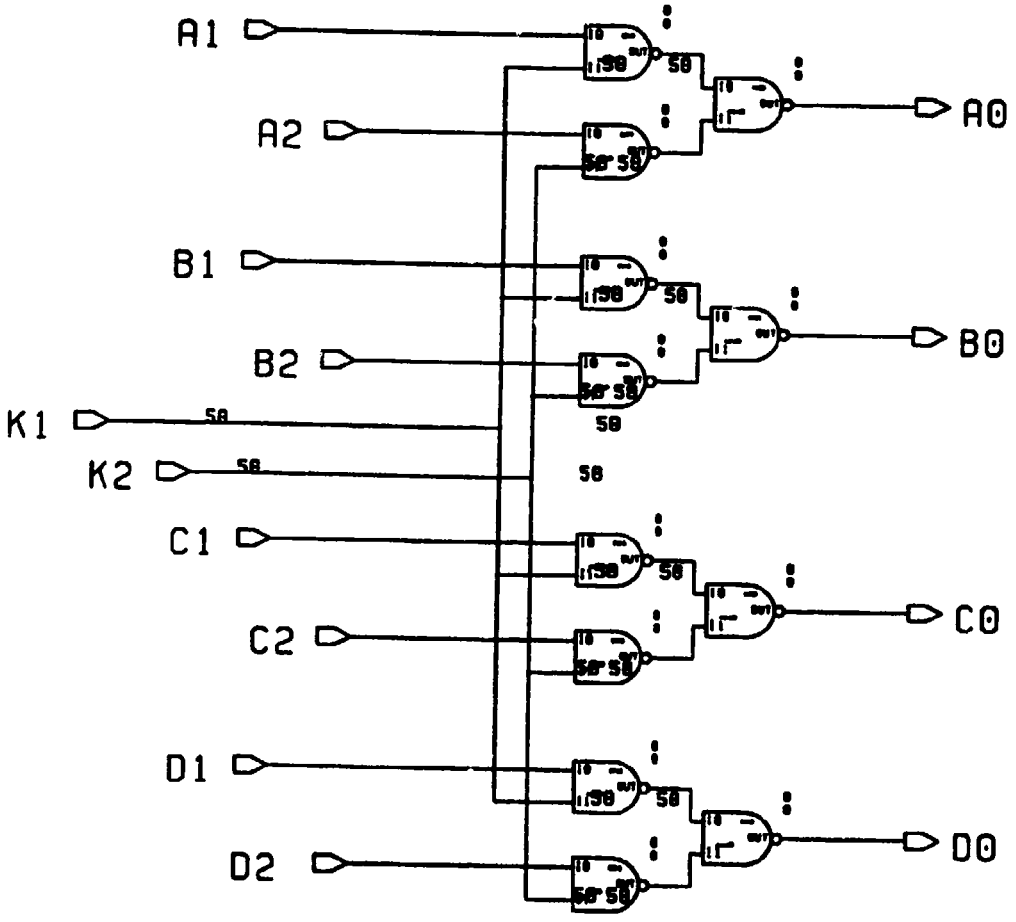


FIGURE 4. AOR Macro Cell

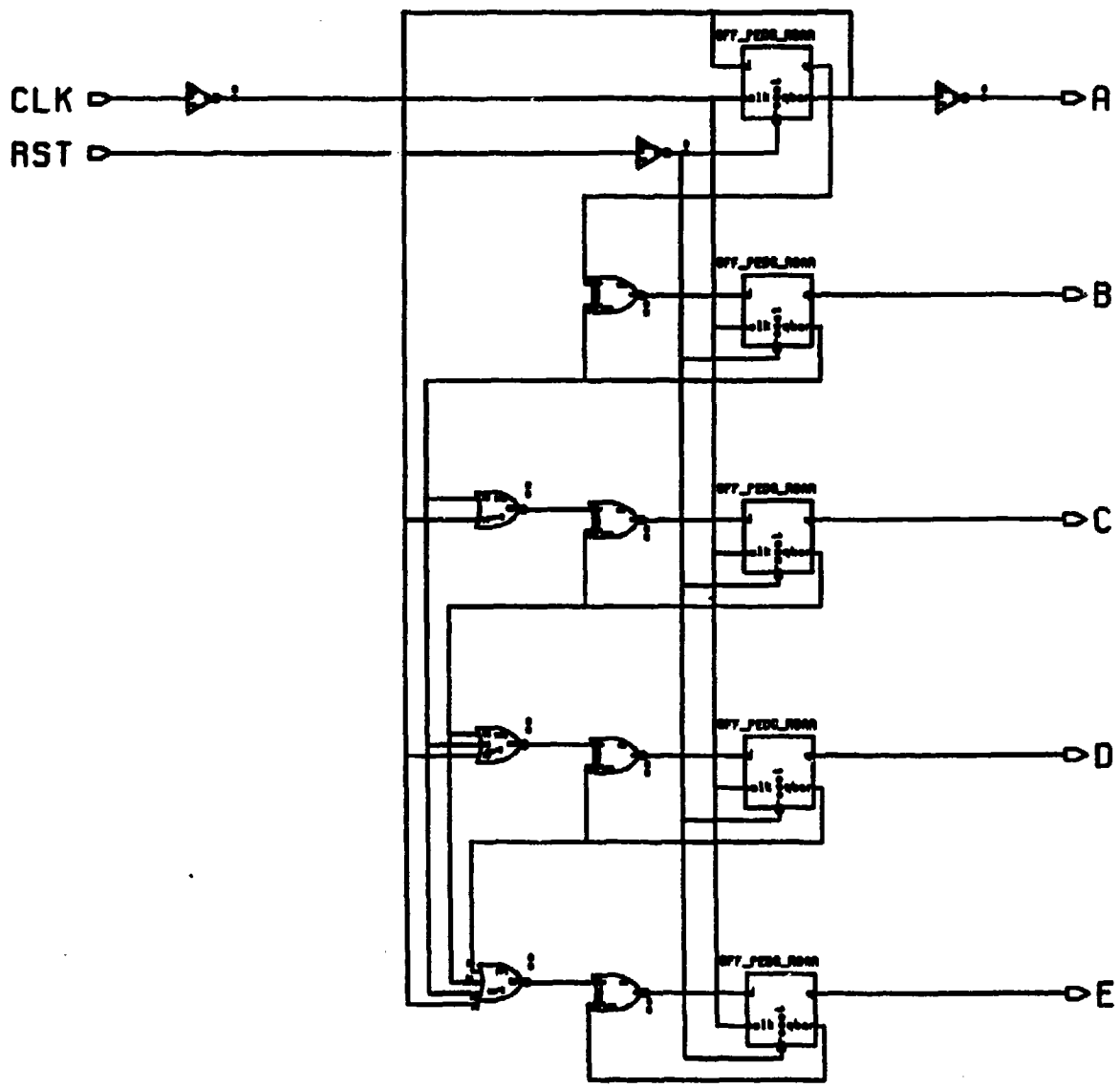


FIGURE 5. Five Bit Up/Down Counter

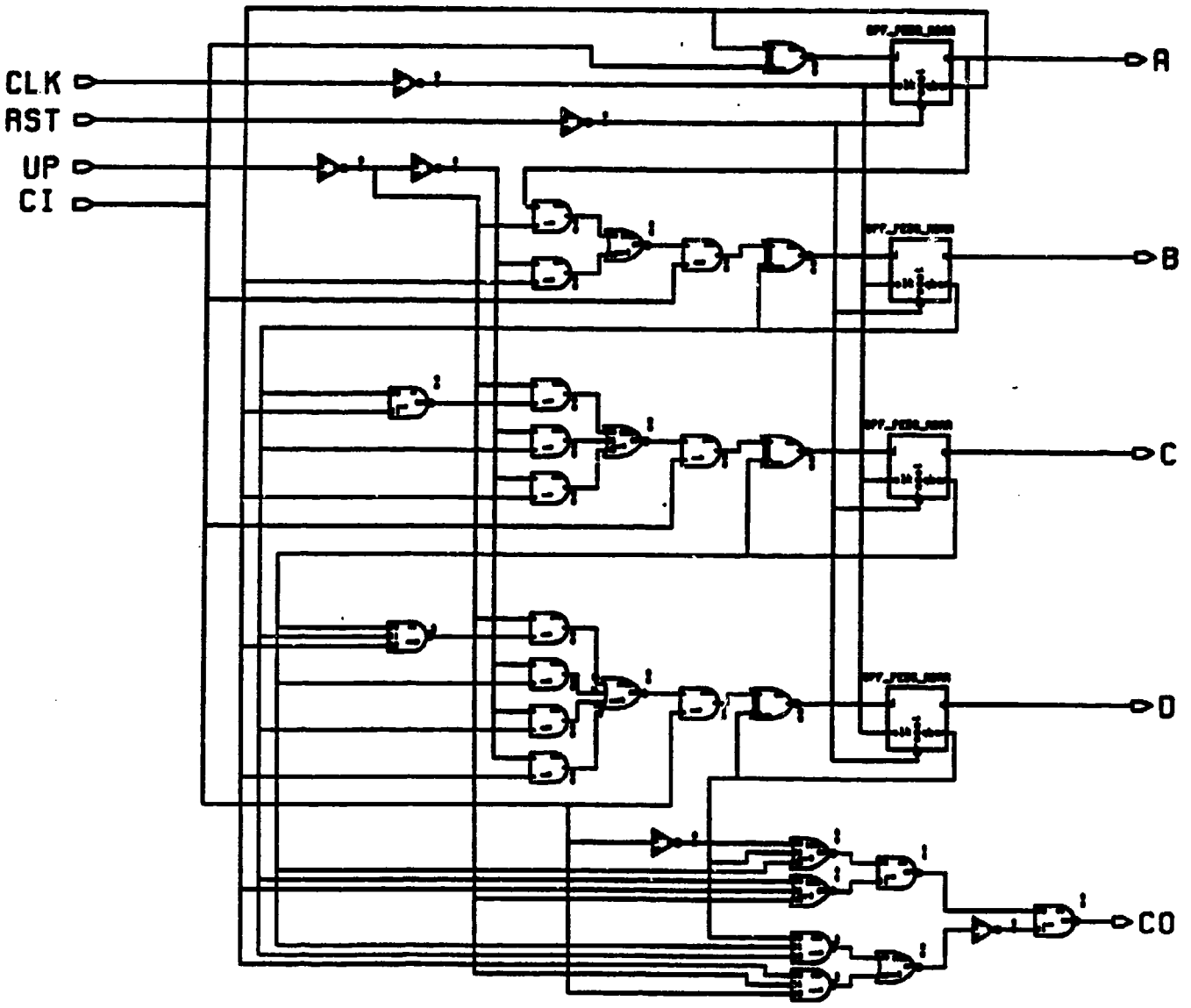


FIGURE 6. Synchronous Four Bit Counter

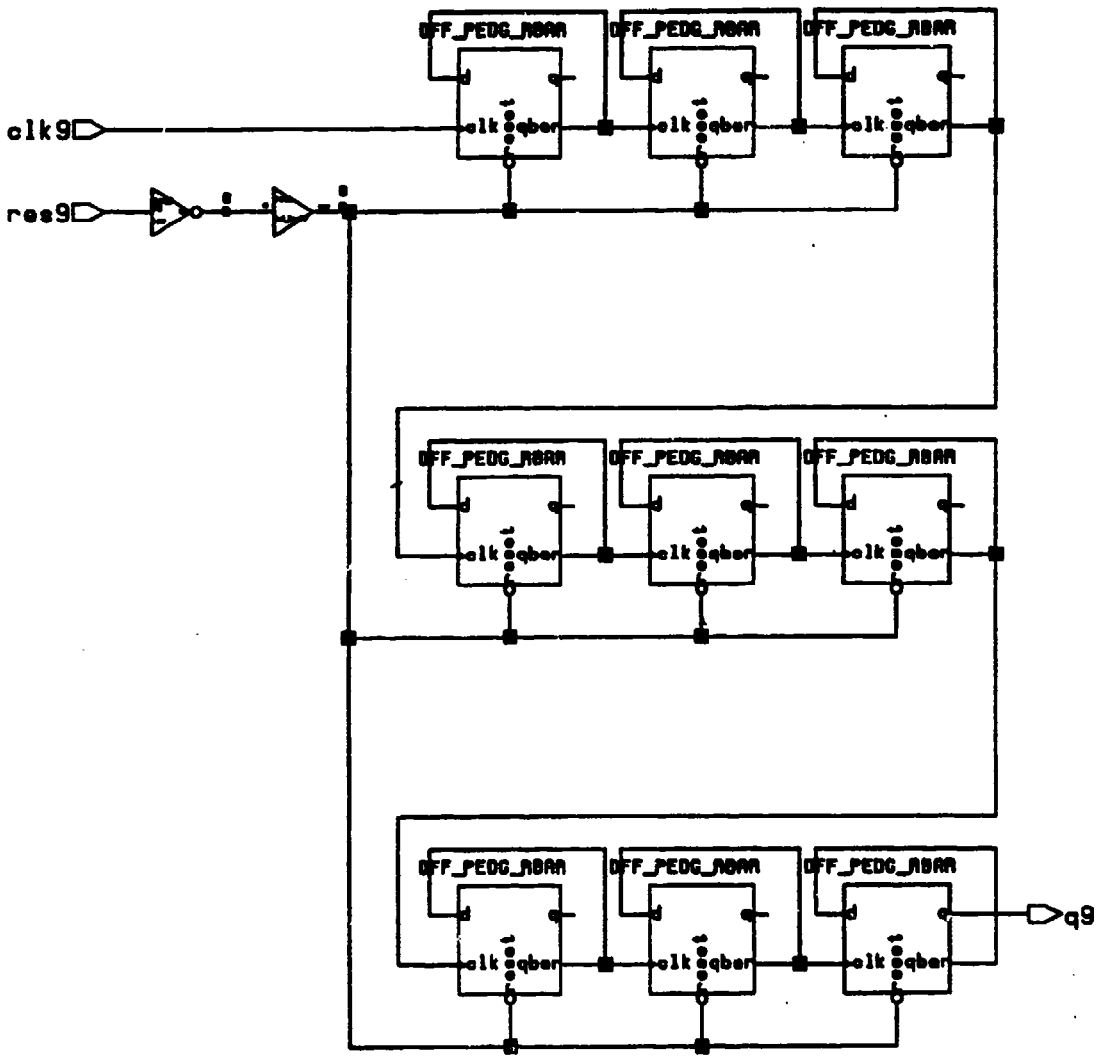


FIGURE 7. Nine Stage Ripple Counter

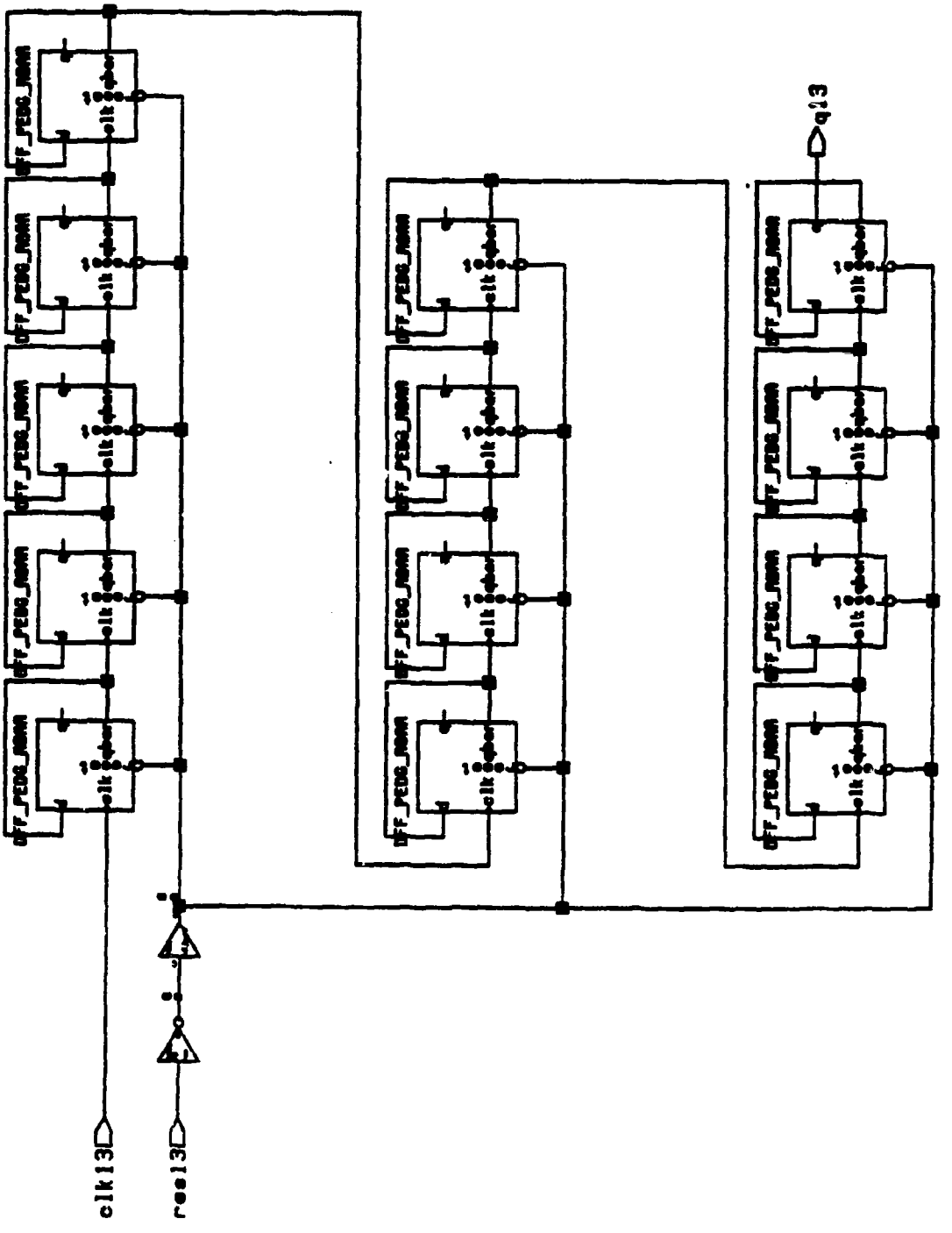


FIGURE 6. Thirteen Stage Ripple Counter

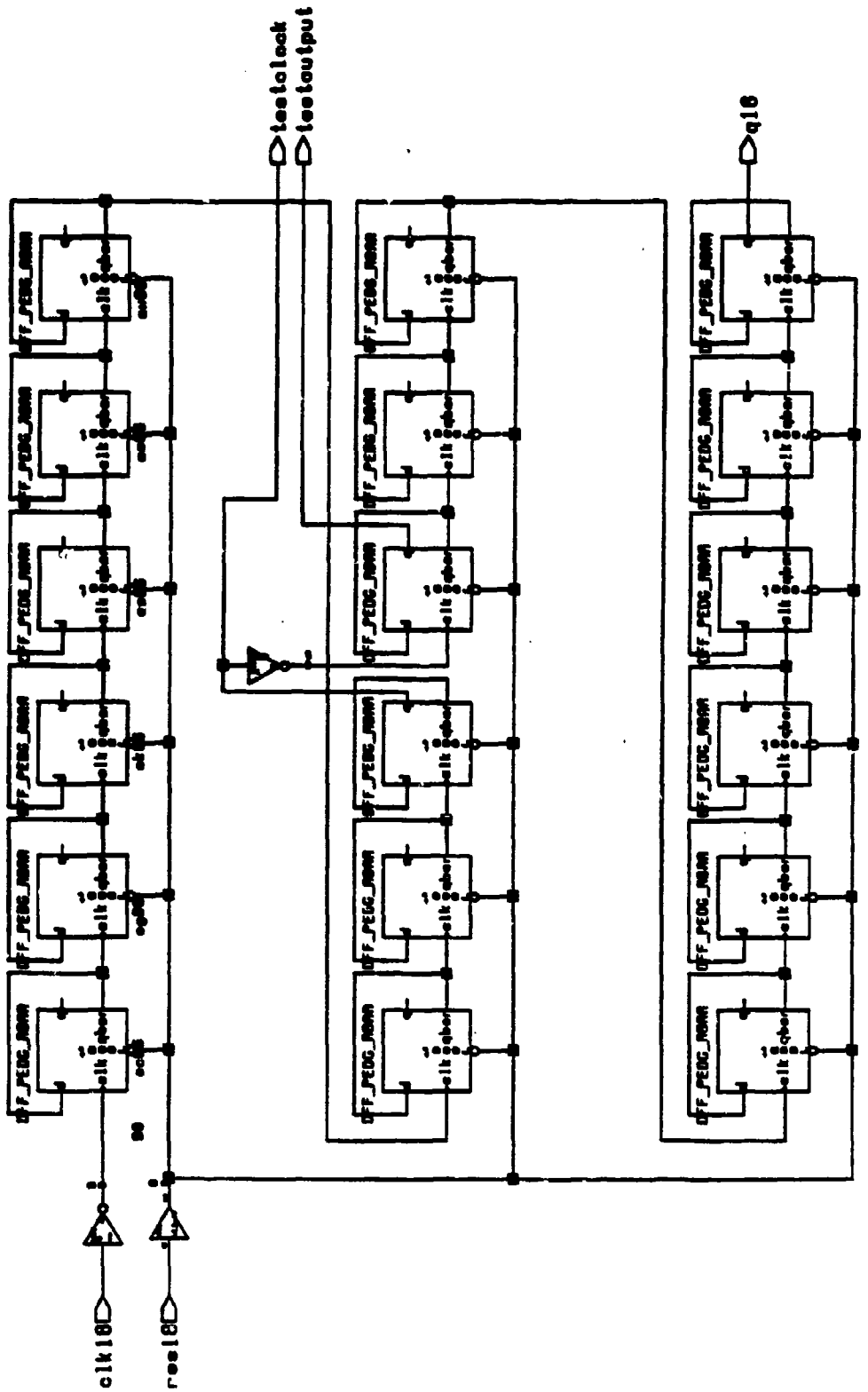


FIGURE 9. Eighteen Stage Ripple Counter

TABLE 1. Macro Cell Instantiations in ASICs Macro Cells

	Macro cells							
GEM cell library element	D flip flop with reset	D flp flop w/ set +reset	and/or register	synchro nous four bit counter	five bit up/ down counter	nine stage ripple counter	thirteen stage ripple counter	18-stage ripple counter
2 input NAND	2		12	3				
3 input NAND	4	6		3				
2 input NOR				2	1			
3 input NOR				3	1			
4 input NOR				1	1			
D flip flop w/ reset				4	5	9	13	18
inverter				6	3	1	1	2
buffer	2	2						
exclu-sive nor				3	3			
hd buffer						1	1	1
2 input AND				12				

Modeling the three ASICs with the proper cell instantiations and signal mapping occurred next (Appendix A). The schematics used to describe the structure were drawn and simulated on a Mentor Graphics workstation. Figures 10 through 12 show the schematics of the ASICs. Table 2 lists the three chips and the numbers of each of the macro cells instantiated in the model of each ASIC. Each ASIC also included several of the basic GEM library cells.

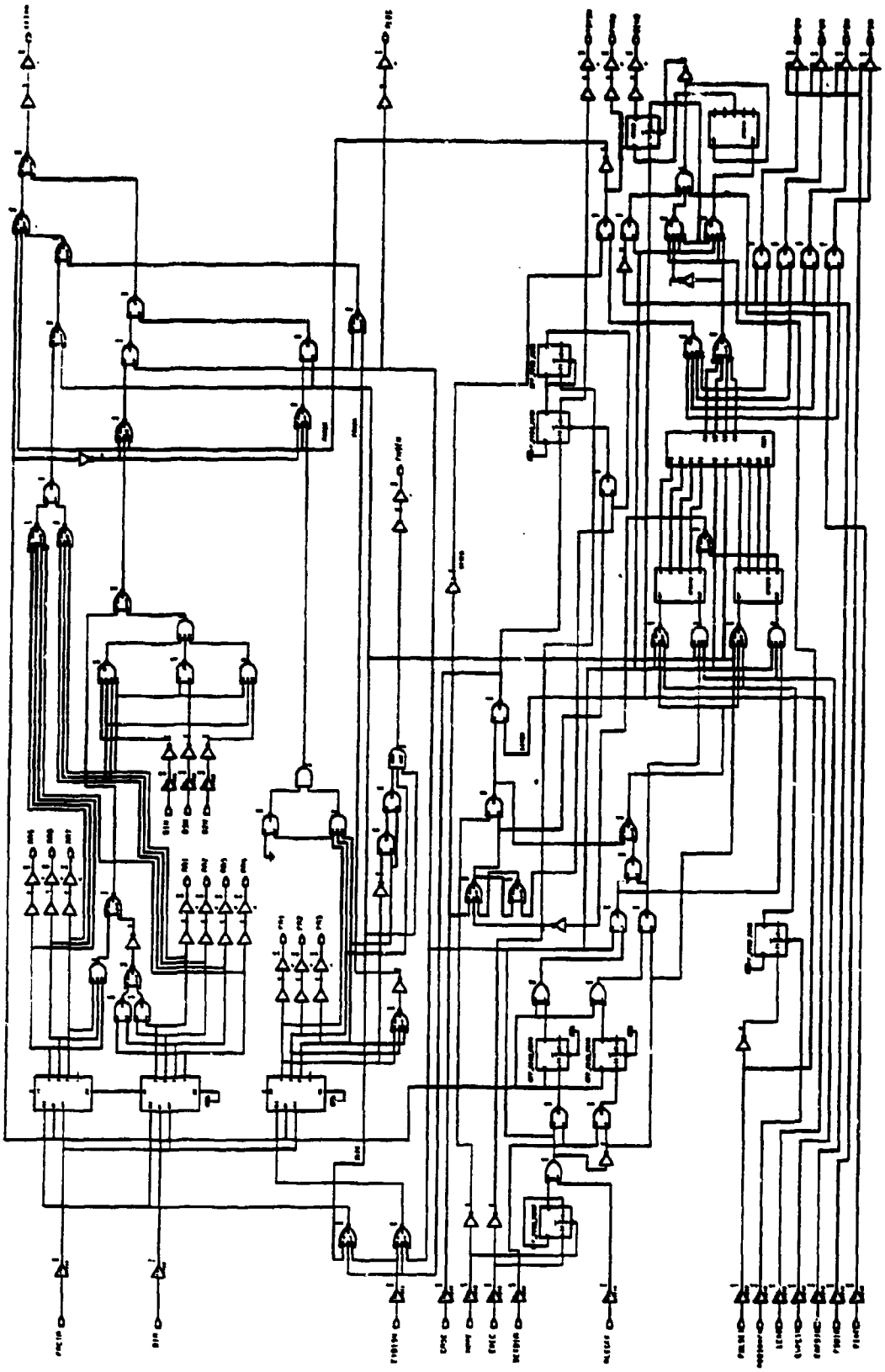


FIGURE 10. Relay and Pulse Counter

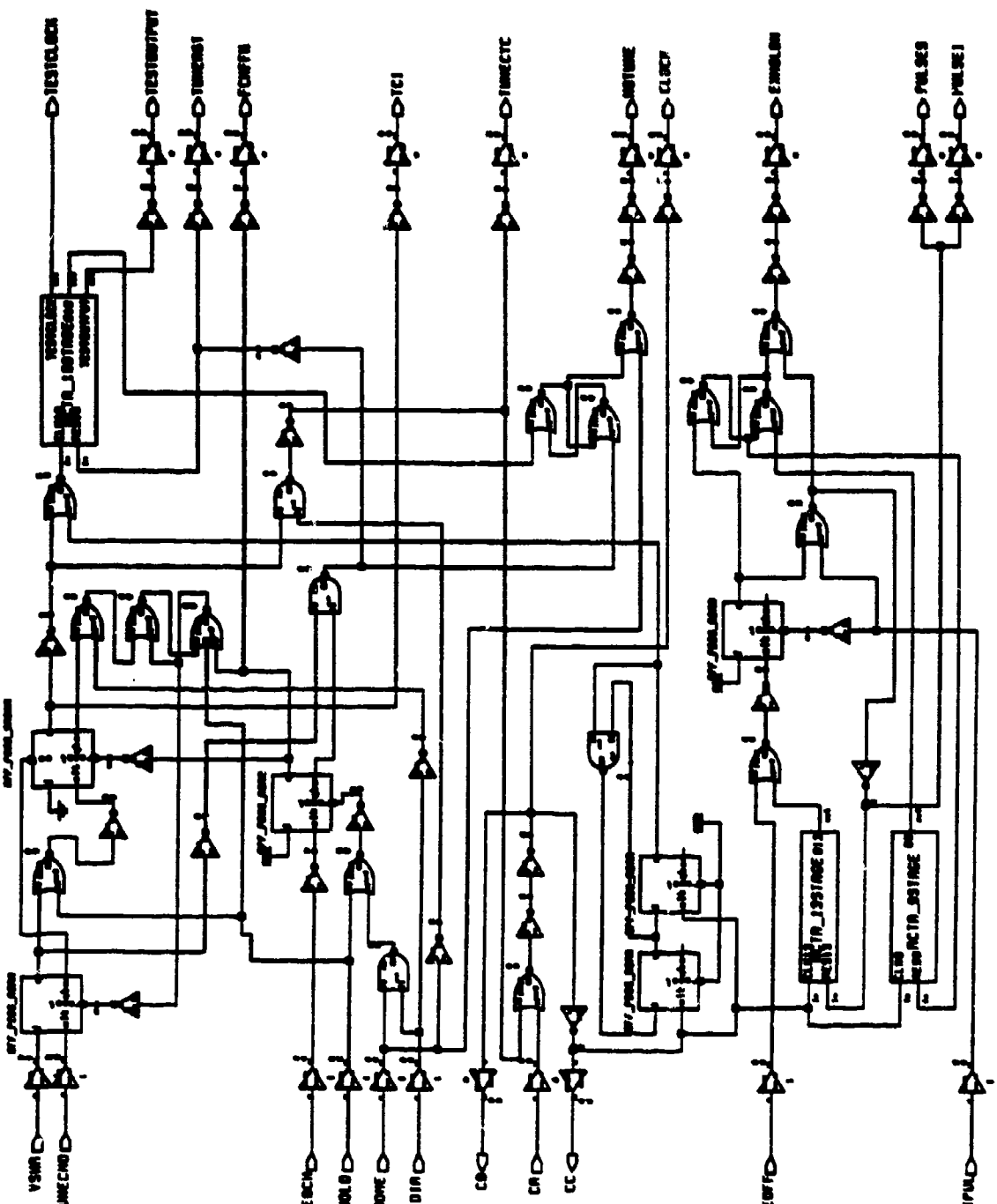


FIGURE 11. Timeout and Delay Counter

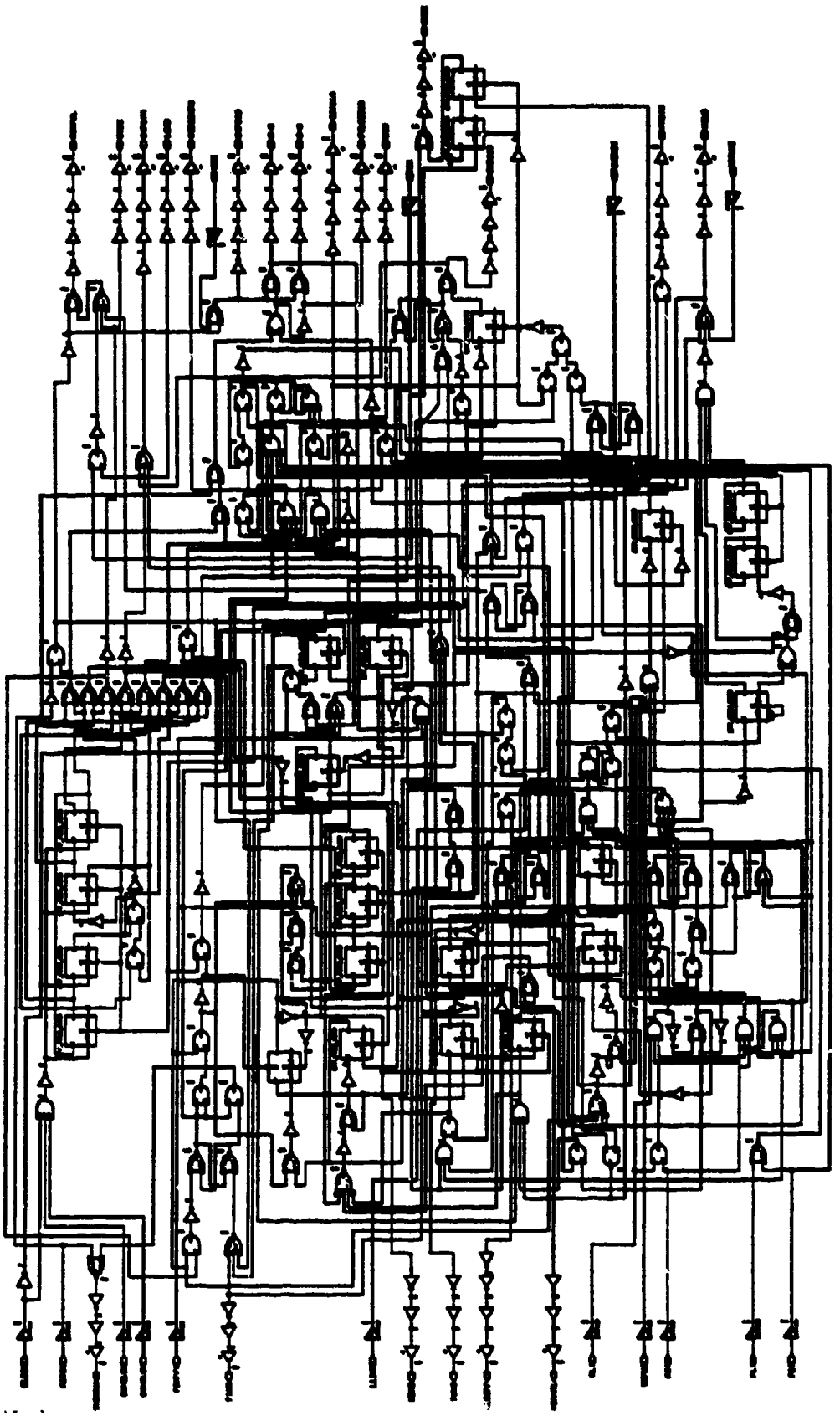


FIGURE 12. Controller Circuit

TABLE 2. Macro Cell Instantiations in ASICs

	Macro cells							
GEM cell library element	**D flip flop with reset**	**D flp flop w/ set+ reset**	**and/or register**	**synchro-nous four bit counter**	**five bit up/ down counter**	**nine stage ripple counter**	**thirteen stage ripple counter**	**18-stage ripple counter**
relay & pulse counter	6		1	3	2			
time-out & delay counter	5	1				1	1	1
control-ler circuit	22	2						

After coding each ASIC description, a small modification file was created for each ASIC. This file was used to pass the operating conditions (voltage and temperature) to the model. Figure 13 diagrams the flow of parameters from the modification file through the hierarchy. The operating conditions were coded into the modification file and were passed as generics through the model hierarchy to the cells and used to determine the cell delays. This method allowed users to change the temperature and voltage conditions easily and quickly without the need to recompile any code below the modification file in the hierarchy. This permitted complete testing of the model throughout the full range of operating conditions (-55°C to 100°C and 9 volts to 11 volts).

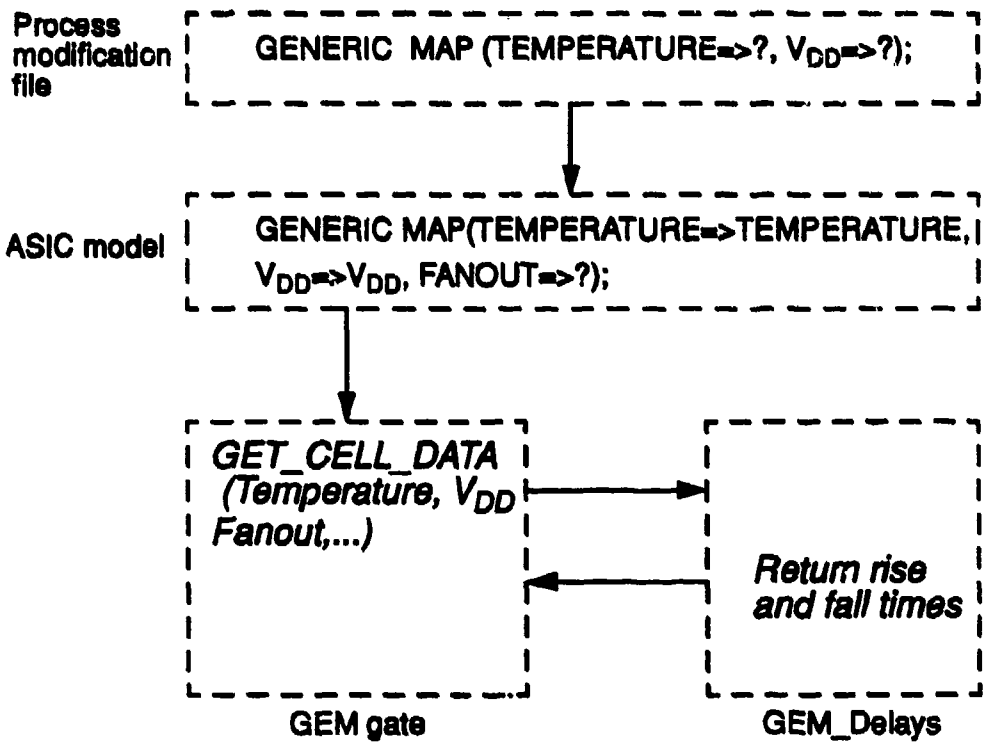


FIGURE 13. Parameter Modification Flow Diagram

After completion of the model of each ASIC, a test bench for each model was created. Each test bench assigned signal values at specified times sent them into the ASIC model through port map statements. The test benches for the relay and pulse counter and the controller circuit models generated simulation run files that contained records of all signal transactions that occurred during the simulation. These run files were used to generate output report files for examination and verification of the simulation. To generate the output report files, report control language files were created which the tool used as a specification for extracting the outputs coded in the run files and formatting them into readable chart files. Each test bench was simulated using both the Intermetrics version 2.2 simulator hosted on a VAX (6000-310) and the Valid (Intermetrics version 3.15) tool set hosted on a SUN Sparcserver 630MP.

Because of the large number of test vectors for the time-out and delay counter, manual validation of the simulation output was not practical. Therefore, a process was added to

the test bench to automate the examination and verification of the simulation results. An ASCII file from the IC tester (pp. 124-125, Appendix A) was used to provide the model's simulation with the appropriate output results for comparison. The lines in the ASCII file were numbered to match line numbers on the truth table supplied as part of the documentation. First the test bench sent input stimuli to the model through hard coded signal assignment statements. After that, the added process controlled the comparison of the output vectors. First, the process read in a line from the ASCII file as a string. Next, the line numbers in the string were converted to time marks. The remainder of the string was divided into input and output strings. Since the inputs for the model were hard-coded, the tester's input string was ignored. The process compiled the output signals at the time marks into another string that was compared with its corresponding output string from the IC tester's ASCII file. Finally, the process wrote both output strings to an ASCII output file (pp. 125-126, Appendix A). The actual converted time mark, a mismatch warning, and the position numbers of any values that did not match between the two output strings were reported after each string of output data in the ASCII output file. This enabled the user to quickly scan the output file for any incorrect output values.

3.0 Problems and Solutions

Functioning circuit designs do not always lead to error-proof VHDL simulations. One particular simulation obstacle to watch for is race conditions in gate pairs that are in a feedback loop. Modeling circuits with gate pairs in feedback loops in an idealized simulation environment can lead to initialization problems. This is due to a race condition in the pair which prevents the outputs of each gate from reaching a stable, defined state. The structural VHDL model of the D flip flop macrocells and pieces of the ASIC models contained such a construct in the form of cross-coupled NAND gates. The outputs of both gates had to initialize to a known state before the simulation could proceed past time zero. However, because the gate delay on each gate was identical, the outputs were continuously changing states during the time zero initialization process. This prevented the simulation from moving past time zero. (Realistically, the feedback paths of a physical circuit will not have exactly identical path delays.) Figure 14 shows a general schematic

of the latch construct and the outputs at time equal to zero when simulated under race conditions.

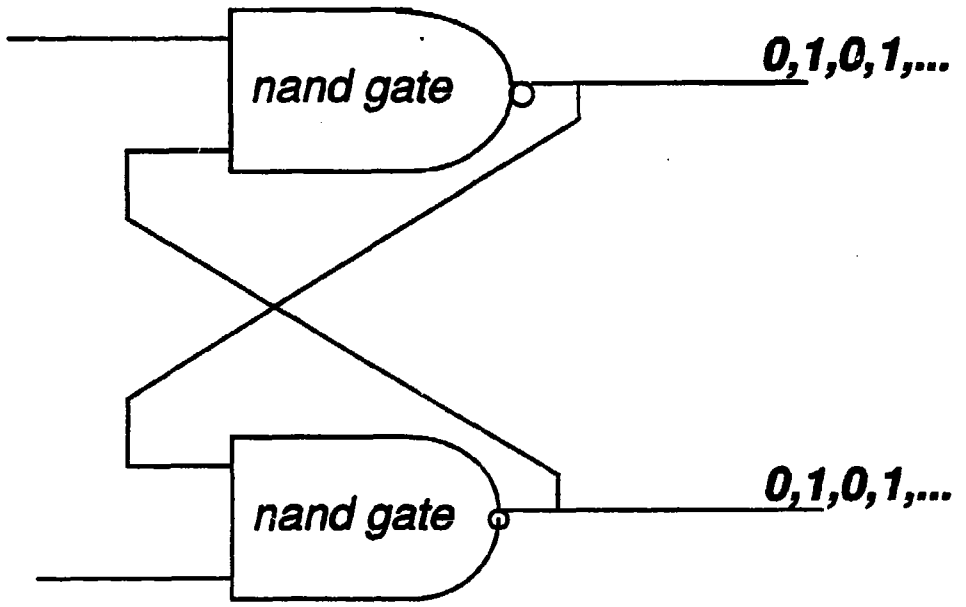


FIGURE 14. Latch Construct With Output States at Time Zero During Simulation

To compensate for this obstacle another generic parameter was added to the NAND gates, AND gates, and NOR gates in the GEM cell library. This parameter, called INITDELAY was used to offset the synchronization of the gate pair. One of the gates of every latch construct was given a non-zero value (1 femtosecond) for INITDELAY, making the path delays unequal between the two gates. This prevented the timing of data through the gates from being completely simultaneous and prevented any simulation race condition in the pair. Figure 3 shows a general schematic of the latch construct and the outputs at time equal to zero when simulated under race conditions.

Another potential obstacle is in verifying the simulation data against IC test data. If a device does not have the appropriate circuitry to power up in a known state, device variation may cause differences in power up state, and therefore in tester results between ICs. This type of device variation is difficult to model in a VHDL simulation. For

instance, the time-out and delay counter contained an eighteen stage ripple counter with a pair of preceding D flip flops functioning as a divide-by-three prescaler (Figure 15).

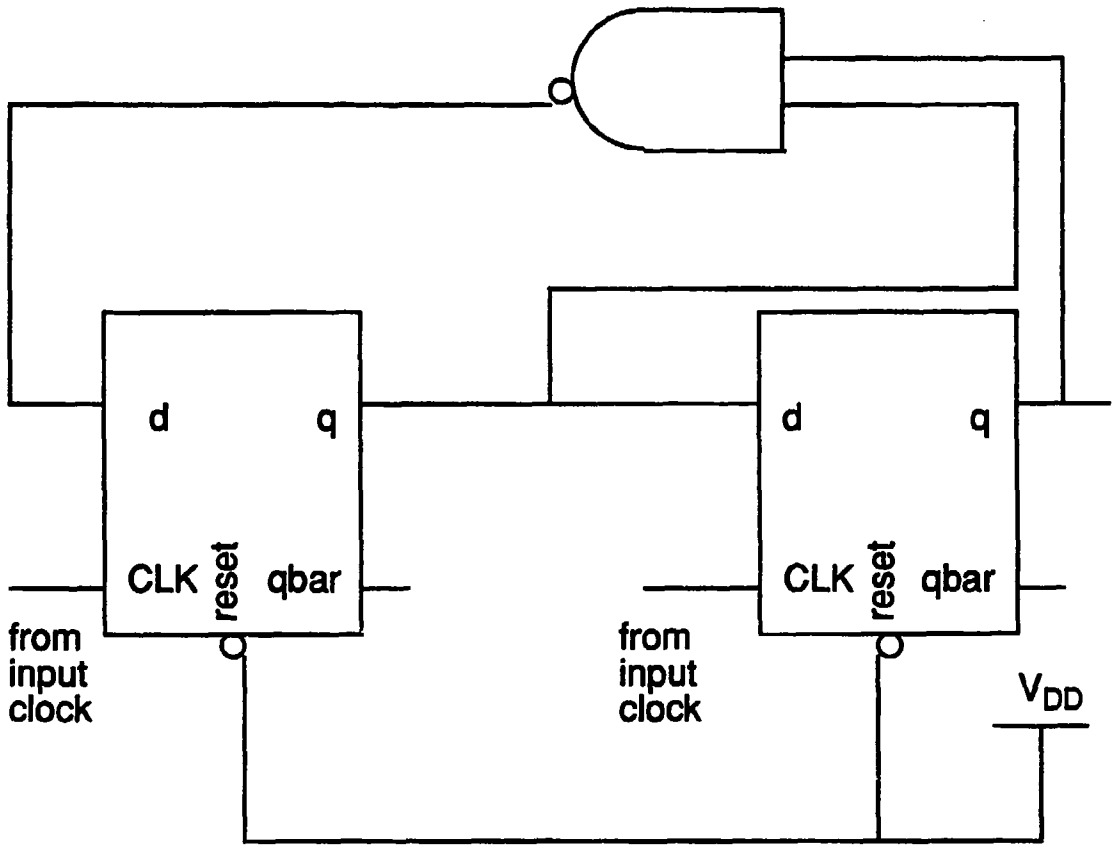


FIGURE 15. Divide-by-Three Prescaler

The active low reset inputs on the two D flip flops were connected to V_{DD}, causing the flip flops to power up in an unknown state. This made it impossible to predict when the ripple counter would start a counting sequence since the q output on the second D flip flop of the prescaler connected to the clock input of the eighteen stage ripple counter. Depending on the initial state, either one, two, or three CLOCK cycles would be necessary to initialize the output to the correct state to start the first counting sequence on the eighteen stage ripple counter. Once the first correct output appears the rest of the operation occurs at regular CLOCK intervals. Therefore, the output sequence is completely predictable after the occurrence of the first correct output. When the circuit was tested on the IC tester, extra cycles were added to the CLOCK input when the first output from the eighteen stage

counter was expected so that there would be one, two, or three CLOCK periods possible when testing the IC. Since it was not predictable which state any given IC would power up in, this allowed one test program to test all of the time-out and delay counter ICs with no additional work. An idealized simulation environment does not identify this problem since the flip flops of the divide by three prescaler will always "power up" in the same default state. Based on the three IC tester results, the VHDL simulation results were matched with the corresponding physical test to determine which test scenario version was being simulated with the original test vectors. Once the correct scenario version was determined, the input vectors in the test bench and the vectors in the ASCII file were changed to correspond to all three scenario versions of the physical IC test vectors. Thus, the model could be verified for accuracy no matter how many CLOCK cycles were necessary to correctly initialize the output of the two D flip flops and activate the eighteen stage ripple counter. Though there are ways of forcing initial states in simulators, there is no easy method of verifying that a non-initialized design works correctly under all conditions. It is advisable to design circuits that have the proper circuitry to ensure that the circuits power up in a known state.

In order to investigate portability of the VHDL models between simulators, the VHDL models were simulated on tools from three different vendors. Several potential portability problems were identified. One problem is the fidelity with which the simulator implements the ANSI/IEEE-1076 standard for VHDL. For example, the first VHDL analyzer used in this effort accepted certain language constructs that did not comply to the language standard. This was only discovered when the models were analyzed in a different vendor's VHDL environment. Another potential difference is that the standard only specifies certain bounds on the range of types: real, integer, and time. A problem that arose while simulating the PRC-70 models on different simulators was that the 1 femtosecond delay selected for INITDELAY caused some simulators to run out of time. A 1-femtosecond time base with the standard's minimum time limit of 2^{31} or 2,147,483,647 base time units gives a maximum simulation time of a little over 2 microseconds. Since each simulation must run for almost 2 milliseconds to pass all of the test vectors, the simulation ran out of time units before simulation was complete. The simulator's error message for this problem was also very difficult to understand.

Simulators with a larger (2^{63}) time limit had no problems with the 1 femtosecond INITDELAY. If the base is one nanosecond, the longest simulation time allowable would be 2.15 seconds. Changing INITDELAY to 1 nanosecond allowed the simulations to complete normally on a simulator with a 2^{31} time base.

4.0 Conclusions

The redesigned ASICs were manufactured using the 1218 cell GEM gate array. The GEM ASICs passed the MIL-STD 883B screening at ARL and were successfully tested at TOAD on the antenna coupler board and in the PRC-70 radio. The PRC-70 ASIC redesign served as an excellent vehicle for studying the utility of VHDL models for documenting, simulating, and validating redesigned ASICs for replacement of hardware composed of obsolete technology and with only paper documentation.

Improvement of the process for design, documentation, and validation is extremely important. Much of the Army's electronic equipment has been in existence for a long time and, in many cases, spare parts are in short supply and are not readily available commercially. The high pace of technology change has caused this difficulty to be a major obstacle in maintaining the operational readiness of equipment. Simulatable human and machine readable documentation, such as VHDL, can play an important role in solving this problem.

5.0 Acknowledgments

The author wishes to gratefully acknowledge the assistance of the following persons during the work discussed herein: Kenneth J. Keyes for the models of the GEM library cells and path delay calculations (tables, gem_constants, and gem_delays), the D flip flop macro cells, the relay and pulse counter and all other macro cells contained in its structure, and the test bench for the relay and pulse counter; West Point Cadet Bryan Tung for the models of the ripple counter macro cells, the time-out and delay counter model, and the special test bench for the time-out and delay counter model; John Erickson for assistance with matching VHDL simulation results with physical test results from the IC tester and

for the ASCII files of IC tester data which were used in the test bench of the time-out and delay counter; and Michael J. McCormick and George Sebesta for their work on designing all the hardware which was modeled in VHDL.

6.0 Bibliography

IEEE Computer Society Standards Committee, "IEEE Standard VHDL Language Reference Manual," ANSI/IEEE Std 1076-1987, New York: IEEE Press 1987.

DI-EGDS-80811. VHISIC Hardware Description Language (VHDL) Documentation. Department of Defense, May 11, 1989.

Lipsett, Roger; Schaefer, Carl F.; Ussery, Cary, *VHDL: Hardware Description and Design*, Boston: Kluwer Academic Publishers 1989.

Perry, Douglas L., *VHDL*, New York: McGraw-Hill, Inc., 1991.

APPENDIX A

VHDL Source Code

```
-- use STD.Simulator_Standard.all;
package tables is

type cell_parameters is record
   Version:        Time;
   Description:    String(1 to 16);
   R:              Integer;
   C:              Real;
   Slope_rise:     Real;
   Intercept_rise: Real;
   Slope_fall:     Real;
   Intercept_fall: Real;
end record;

type cell_values is array (Positive range <>) of cell_parameters;

constant inv_table :       cell_values:= (1 =>
( 1 ns,  "inv_1ns      ",  0, 0.0, 5.60, 8.00, 7.00, 8.00)
                           );

constant inv_fast_table :  cell_values:= (1 =>
( 1 ns,  "inv_fast_1ns ",  0, 0.0, 5.60, 8.00, 7.00, 8.00)
                           );

constant and2_table       : cell_values:= (1 =>
( 1 ns,  "and2_1ns     ",  0, 0.0, 6.80, 9.40, 7.30, 7.90)
                           );

constant and2_fast_table  : cell_values:= (1 =>
( 1 ns,  "and2_fast_1ns ", 0, 0.0, 6.80, 9.40, 7.30, 7.90)
                           );

constant and3_table       : cell_values:= (1 =>
( 1 ns,  "and3_1ns     ",  0, 0.0, 7.00, 15.3, 7.60, 8.00)
                           );

constant and3_fast_table  : cell_values:= (1 =>
( 1 ns,  "and3_fast_1ns ", 0, 0.0, 7.00, 15.3, 7.60, 8.00)
                           );

constant nand2_table :     cell_values:= (1 =>
( 1 ns,  "nand2_1ns    ",  0, 0.0, 5.70, 8.10, 7.00, 11.6)
                           );

constant nand2_fast_table : cell_values:= (1 =>
( 1 ns,  "nand2_fast_1ns ", 0, 0.0, 5.70, 8.10, 7.00, 11.6)
                           );

constant nand3_table :     cell_values:= (1 =>
( 1 ns,  "nand3_1ns    ",  0, 0.0, 5.80, 8.20, 5.60, 22.6)
                           );

constant nand3_fast_table : cell_values:= (1 =>
( 1 ns,  "nand3_fast_1ns ", 0, 0.0, 5.80, 8.20, 5.60, 22.6)
```

```
                        );

constant nand4_table :     cell_values:= (1 =>
( 1 ns,   "nand4_1ns     ",   0,  0.0,  6.00,  8.70,  6.00,  29.4)
                        );

constant nand4_fast_table : cell_values:= (1 =>
( 1 ns,   "nand4_fast_1ns ",   0,  0.0,  6.00,  8.70,  6.00,  29.4)
                        );

constant nand5_table :     cell_values:= (1 =>
( 1 ns,   "nand5_1ns     ",   0,  0.0,  5.80,  8.30,  7.00,  30.9)
                        );

constant nand5_fast_table : cell_values:= (1 =>
( 1 ns,   "nand5_fast_1ns ",   0,  0.0,  5.80,  8.30,  7.00,  30.9)
                        );

constant nor2_table :      cell_values:= (1 =>
( 1 ns,   "nor2_1ns      ",   0,  0.0,  5.90,  12.1,  7.30,  8.10)
                        );

constant nor2_fast_table :  cell_values:= (1 =>
( 1 ns,   "nor2_fast_1ns  ",   0,  0.0,  5.90,  12.1,  7.30,  8.10)
                        );

constant nor3_table :      cell_values:= (1 =>
( 1 ns,   "nor3_1ns      ",   0,  0.0,  4.30,  22.2,  7.30,  8.10)
                        );

constant nor3_fast_table :  cell_values:= (1 =>
  1 ns,   "nor3_fast_1ns  ",   0,  0.0,  4.30,  22.2,  7.30,  8.10)
                        );

constant nor4_table :      cell_values:= (1 =>
( 1 ns,   "nor4_1ns      ",   0,  0.0,  5.10,  2.30,  7.30,  8.10)
                        );

constant nor4_fast_table :  cell_values:= (1 =>
( 1 ns,   "nor4_fast_1ns  ",   0,  0.0,  5.10,  2.30,  7.30,  8.10)
                        );

constant nor5_table :      cell_values:= (1 =>
( 1 ns,   "nor5_1ns      ",   0,  0.0,  5.20,  2.93,  7.40,  8.20)
                        );

constant nor5_fast_table :  cell_values:= (1 =>
( 1 ns,   "nor5_fast_1ns  ",   0,  0.0,  5.20,  2.93,  7.40,  8.20)
                        );

constant exor_table :      cell_values:= (1 =>
( 1 ns,   "exor_1p1ns    ",   0,  0.0,  4.00,  16.0,  6.80,  12.0)
                        );
```

```
constant exor_fast_table :  cell_values:= (1 =>
( 1 ns,  "exor_fast     ",  0,  0.0,  4.00,  16.0,  6.80,  12.0)
                            );

constant exnor_table :      cell_values:= (1 =>
( 1 ns,  "exnor_1ns     ",  0,  0.0,  7.00,  10.0,  7.20,  10.0)
                            );

constant exnor_fast_table : cell_values:= (1 =>
( 1 ns,  "exnor_fast    ",  0,  0.0,  7.00,  10.0,  7.20,  10.0)
                            );

constant rc_buff_table :    cell_values:= (1 =>
( 1 ns,  "rc_buff       ",  0,  0.0,  0.82,  1.40,  0.87,  3.20)
                            );

constant rc_buff_f_table :  cell_values:= (1 =>
( 1 ns,  "rc_buff_f     ",  0,  0.0,  0.82,  1.40,  0.87,  3.20)
                            );

constant buff_table :       cell_values:= (1 =>
( 1 ns,  "hd_buff_1ns   ",  0,  0.0,  4.00,  8.30,  5.00,  7.80)
                            );

constant out_buff_table :   cell_values:= (1 =>
( 1 ns,  "out_buff_1ns  ",  0,  0.0,  5.60,  8.00,  7.00,  8.00)
                            );

constant hd_buff_table :    cell_values:= (1 =>
( 1 ns,  "hd_buff_1ns   ",  0,  0.0,  4.00,  8.30,  5.00,  7.80)
                            );

constant cmos_in_table :    cell_values:= (1 =>
( 1 ns,  "cmos_in_1ns   ",  0,  0.0,  0.10,  1.10,  0.10,  1.10)
                            );

constant cmos_out_table :   cell_values:= (1 =>
( 1 ns,  "cmos_out_1ns  ",  0,  0.0,  5.60,  8.00,  7.00,  8.00)
                            );

constant cmos_pad_table :   cell_values:= (1 =>
( 1 ns,  "cmos_pad_1ns  ",  0,  0.0,  5.60,  8.00,  7.00,  8.00)
                            );

constant cmos_5vout_table : cell_values:= (1 =>
( 1 ns,  "cmos_5vout_1ns ", 0,  0.0,  5.60,  8.00,  7.00,  8.00)
                            );

constant aoi22_table :      cell_values:= (1 =>
( 2 ns,  "aoi22_2ns     ",  0,  0.0,  0.25,  1.25,  0.37,  1.25)
                            );
```

```
constant aoi322b_table :   cell_values:= (1 =>
( 2 ns,   "aoi322b_2ns   ",   0,  0.0,  0.25,  1.25,  0.37,  1.25)
                           );

constant aoi332b_table :   cell_values:= (1 =>
( 2 ns,   "aoi332b_2ns   ",   0,  0.0,  0.25,  1.25,  0.37,  1.25)
                           );

end tables;


-- use STD.Simulator_Standard.all;
package GEM_CONSTANTS is
-------------------------------------------------------------
-- The "LOAD" constant is the Input Load Capacitance of each Input
-- Pin of each gate in the GEM Library. This "LOAD" constant is
-- used to calculate the Fanout Delay with respect to the number
-- of gates being driven by a single Output.

constant LOAD:REAL:=0.15;
-------------------------------------------------------------
end GEM_CONSTANTS;

-- use STD.Simulator_Standard.all;

use work.GEM_CONSTANTS.all;
use work.TABLES.all;

-------------------------------------------------------------
-- Package specification

package GEM_DELAYS is

  type tri_state is ('X', '0', '1');

  signal cell_version:string(1 to 16);
  signal rise_d,fall_d:real;
  signal risedelay,falldelay:time;

  function GET_LOAD_DATA(cell_table:cell_values;DELAY:time)
     return cell_parameters;
  function GET_LOAD_DELAY(SLOPE:real;INTERCEPT:real;LOAD:real;
                  FANOUT:integer) return real;

  function VOLTAGE_DELAY_R(VDD:real) return real;
  function VOLTAGE_DELAY_F(VDD:real) return real;
  function TEMPERATURE_DELAY_R(DELAY:real;TEMPERATURE:real) return real;
  function TEMPERATURE_DELAY_F(DELAY:real;TEMPERATURE:real) return real;

  function RISE_DELAY(VDD:real;
  LOAD_DEL:real;
  TEMPERATURE:real) return time;
  function FALL_DELAY(VDD:real;
  LOAD_DEL:real;
```

```
 TEMPERATURE:real) return time;

procedure GET_CELL_DATA(cell_table: in cell_values; DELAY: in time;
                LOAD,VDD,TEMPERATURE:  in real;
                FANOUT:  in integer;
                signal cell_version:  out string(1 to 16);
                signal rise_d,fall_d:  out real;
                signal RISEDELAY,FALLDELAY:  out time);


end GEM_DELAYS;



-----------------------------------------------------------------
-----------------------------------------------------------------
-----------------------------------------------------------------
-- Package Body

package body GEM_DELAYS is

-----------------------------------------------------------------

 function GET_LOAD_DATA(cell_table:cell_values;DELAY:time)
    return cell_parameters is

   variable parameters:cell_parameters;

   begin
for i in cell_table'Range loop
   if cell_table(i).Version = DELAY then
parameters := cell_table(i);
return parameters;
   end if;
end loop;

assert false
Report "DELAY value not found in cell table."
Severity ERROR;

 end GET_LOAD_DATA;


 function GET_LOAD_DELAY(SLOPE:real;INTERCEPT:real;LOAD:real;
       FANOUT:integer) return real is

   variable LOAD_DEL:real;
   constant FANOUT_MAX:  integer :=40;

   begin

     if (FANOUT < FANOUT_MAX) then
LOAD_DEL := real(FANOUT)*(SLOPE * LOAD) + INTERCEPT ;
```

```
else
 LOAD_DEL := real(FANOUT_MAX)*(SLOPE*LOAD) + INTERCEPT;
   end if;

   assert ( FANOUT > 0 )
report "The fanout is 0 or negative; fanout delay defaults to 0 ns."
severity WARNING;

   assert (FANOUT <= FANOUT_MAX)
report "The fanout is greater than 40; it defaults to 40 gate loads."
severity WARNING;


return LOAD_DEL;
  end GET_LOAD_DELAY;

-------------------------------------------------------------
-------------------------------------------------------------
-- function VOLTAGE_DELAY
-- This function calculates the basic propagation delay of the gate
-- given VDD.
-------------------------------------------------------------


 function VOLTAGE_DELAY_R(VDD:real) return real is

  variable TEMP_DELAY_R:  real;
  constant VDD_MIN: real:= 9.0;
  constant VDD_MAX: real:= 11.0;

  begin

if (VDD < VDD_MIN) then
 TEMP_DELAY_R := (0.64788
         + (0.070739*(1.0/VDD_MIN))
         + (8.4494*(1.0/VDD_MAX)*(1.0/VDD_MIN))) ;

else if (VDD > VDD_MAX) then
 TEMP_DELAY_R := (0.64788
         + (0.070739*(1.0/VDD_MAX))
         + (8.4494*(1.0/VDD_MAX)*(1.0/VDD_MAX))) ;

else
 TEMP_DELAY_R := (0.64788
         + (0.070739*(1.0/VDD))
         + (8.4494*(1.0/VDD)*(1.0/VDD))) ;

   end if;
end if;


if (TEMP_DELAY_R < 0.0) then
      return 0.0;
end if;
```

```
   assert (TEMP_DELAY_R >= 0.0)
report "Calculated voltage delay is less than zero; zero is used."
severity WARNING;

   assert (VDD >= VDD_MIN)
report "VDD less than minimum; VDD defaults to VDD minimum."
severity WARNING;

   assert (VDD <= VDD_MAX)
report "VDD greater than maximum; VDD defaults to VDD maximum."
severity WARNING;


   return TEMP_DELAY_R;

  end VOLTAGE_DELAY_R;


 function VOLTAGE_DELAY_F(VDD:real) return real is

  variable TEMP_DELAY_F:  real;
  constant VDD_MIN: real:= 9.0;
  constant VDD_MAX: real:= 11.0;

  begin

if (VDD < VDD_MIN) then
 TEMP_DELAY_F := (1.9983
               - (8.6695*(1.0/VDD_MIN))
               + (18.394*(1.0/VDD_MIN)*(1.0/VDD_MIN))) ;

else if (VDD > VDD_MAX) then
 TEMP_DELAY_F := (1.9983
               - (8.6695*(1.0/VDD_MAX))
               + (18.394*(1.0/VDD_MAX)*(1.0/VDD_MAX))) ;


else
 TEMP_DELAY_F := (1.9983
               - (8.6695*(1.0/VDD))
               + (18.394*(1.0/VDD)*(1.0/VDD))) ;

   end if;
end if;

   if (TEMP_DELAY_F < 0.0) then
     return 0.0;
   end if;

   assert (TEMP_DELAY_F >= 0.0)
report "Calculated voltage delay is less than zero; zero is used."
severity WARNING;
```

```
   assert (VDD >= VDD_MIN)
report "VDD less than minimum; VDD defaults to VDD minimum."
severity WARNING;

   assert (VDD <= VDD_MAX)
report "VDD greater than maximum; VDD defaults to VDD maximum."
severity WARNING;

   return TEMP_DELAY_F;

  end VOLTAGE_DELAY_F;

-------------------------------------------------------------
-------------------------------------------------------------
-- function Temperature_Delay
-- This function compensates for delays other than 298.15 degrees K
-- by a percent specified by constant TEMPERATURE_COEFF.
--
-- For temperatures above 398.15 degrees K, the delay at 398.15 degrees K
-- is returned. Likewise for temperatures below 0 degrees K.
-------------------------------------------------------------


 function TEMPERATURE_DELAY_R(DELAY:real;TEMPERATURE:real) return real is


   variable TEMP_DELAY_R:  real;
   constant TEMP_MIN: real:= 0.0;
   constant TEMP_MAX: real:= 398.15;
   begin

if (TEMPERATURE < TEMP_MIN) then
 TEMP_DELAY_R := DELAY*(0.52246
+ ((1.23E-3)*TEMP_MIN)
+ ((1.25E-6)*TEMP_MIN*TEMP_MIN)) ;

 else if (TEMPERATURE > TEMP_MAX) then
  TEMP_DELAY_R := DELAY*(0.52246
          + ((1.23E-3)*TEMP_MAX)
          + ((1.25E-6)*TEMP_MAX*TEMP_MAX)) ;

else
 TEMP_DELAY_R := DELAY*(0.52246
         + ((1.23E-3)*TEMPERATURE)
         + ((1.25E-6)*TEMPERATURE*TEMPERATURE)) ;
 end if;
end if;

    if (TEMP_DELAY_R < 0.0 ) then
return 0.0;
    end if;

   assert (TEMP_DELAY_R >= 0.0)
report "Calculated temperature delay is less than zero; zero is used."
```

```
severity WARNING;

    assert (TEMPERATURE >= 0.0)
report "Temperature specified less than 0; 0 is used."
severity WARNING;

    assert (TEMPERATURE <= 378.15)
report "Temperature specified greater than 398.15; 398.15 is used."
severity WARNING;


    return TEMP_DELAY_R;

 end TEMPERATURE_DELAY_R;


 function TEMPERATURE_DELAY_F(DELAY:real;TEMPERATURE:real) return real is

  variable TEMP_DELAY_F:  real;
  constant TEMP_MIN: real:= 0.0;
  constant TEMP_MAX: real:= 398.15;

  begin

if (TEMPERATURE < TEMP_MIN) then
 TEMP_DELAY_F := DELAY*(0.50673
          + ((1.6967E-3)*TEMP_MIN)
          - ((1.3889E-7)*TEMP_MIN*TEMP_MIN)) ;

 else if (TEMPERATURE > TEMP_MAX) then
  TEMP_DELAY_F := DELAY*(0.50673
           + ((1.6967E-3)*TEMP_MAX)
           - ((1.3889E-7)*TEMP_MAX*TEMP_MAX)) ;

else
 TEMP_DELAY_F := DELAY*(0.50673
          + ((1.6967E-3)*TEMPERATURE)
          - ((1.3889E-7)*TEMPERATURE*TEMPERATURE)) ;

 end if;
end if;

  if (TEMP_DELAY_F < 0.0) then
    return 0.0;
  end if;


    assert (TEMP_DELAY_F >= 0.0)
report "Calculated temperature delay is less than zero; zero is used."
severity WARNING;

    assert (TEMPERATURE >= 0.0)
report "Temperature specified less than 0; 0 is used."
severity WARNING;
```

```
    assert (TEMPERATURE <= 378.15)
report "Temperature specified greater than 398.15; 398.15 is used."
severity WARNING;

    return TEMP_DELAY_F;

 end TEMPERATURE_DELAY_F;

-----------------------------------------------------------------
-----------------------------------------------------------------
-- function RISE_DELAY
--
-- This function combines the voltage delay, temperature delay,
-- fanout, and rise/fall time ratio to calculate the propagation time
-- to a one output.
--
-----------------------------------------------------------------

function RISE_DELAY(VDD:real;
LOAD_DEL:real;
TEMPERATURE:real) return time is

variable TEMP_DELAY_R:real;
variable TEMP2_DELAY:real;
variable TOTAL_DELAY:real;

begin

 TEMP_DELAY_R := VOLTAGE_DELAY_R(VDD) * LOAD_DEL ;
 TEMP2_DELAY  := TEMPERATURE_DELAY_R(TEMP_DELAY_R,TEMPERATURE);
 TOTAL_DELAY  := TEMP2_DELAY;

 return integer(TOTAL_DELAY) * 1 ns;

end RISE_DELAY;

-----------------------------------------------------------------
-----------------------------------------------------------------
-- function FALL_DELAY
--
-- This function combines the voltage delay, temperature delay,
-- fanout, and rise/fall time ratio to calculate the propagation time
-- to a zero output.
--
-----------------------------------------------------------------

function FALL_DELAY(VDD:real;
   LOAD_DEL:real;
   TEMPERATURE:real) return time is

variable TEMP_DELAY_F:real;
variable TEMP2_DELAY:real;
variable TOTAL_DELAY:real;
```

```
begin

 TEMP_DELAY_F := VOLTAGE_DELAY_F(VDD) * LOAD_DEL ;
 TEMP2_DELAY  := TEMPERATURE_DELAY_F(TEMP_DELAY_F,TEMPERATURE);
 TOTAL_DELAY  := TEMP2_DELAY;

 return integer(TOTAL_DELAY) * 1 ns;

end FALL_DELAY;

-------------------------------------------------------------------
-------------------------------------------------------------------
-- procedure GET_CELL_DATA
-- This procedure gets all the necessary parameters, from all files,
-- that are used to calculate the propagation delays.
--

procedure GET_CELL_DATA(cell_table: in cell_values; DELAY: in time;
                LOAD,VDD,TEMPERATURE: in real;
                FANOUT: in integer;
                signal cell_version: out String(1 to 16);
                signal rise_d,fall_d: out real;
                signal RISEDELAY,FALLDELAY: out time) is

variable rised,falld:real;
variable cell:cell_parameters;

begin

   wait for 1 ns;
   cell := GET_LOAD_DATA(cell_table,DELAY);
   rised := GET_LOAD_DELAY(cell.Slope_rise,cell.Intercept_rise,LOAD,FANOUT);
   falld := GET_LOAD_DELAY(cell.Slope_fall,cell.Intercept_fall,LOAD,FANOUT);
   cell_version <= cell.Description;
   rise_d <= rised;
   fall_d <= falld;
   RISEDELAY <= RISE_DELAY(VDD,
                rised,
                TEMPERATURE);
   FALLDELAY <= FALL_DELAY(VDD,
                falld,
                TEMPERATURE);

   wait;

end GET_CELL_DATA;
-------------------------------------------------------------------
-------------------------------------------------------------------

end GEM_DELAYS;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;

-------------------------------
-- device AND2
-- file and2.vhd
-- 2 Input AND Gate
-------------------------------

entity AND2 is

 generic (DELAY:time;
--        LOAD:real;
        FANOUT:integer;
        VDD:real;
        TEMPERATURE:real;
        AND2_TABLE:cell_values);

 port (INPUT1 : in bit;
     INPUT2 : in bit;
     OUTPUT : out bit);

end AND2;


architecture BEHAVIORAL of AND2 is

--
-- Functions Rise_delay and Fall_delay calculate the delay for
-- the voltage, fanout and temperature of a given gate instant
--

signal cell_version: String(1 to 16):="               ";
signal rise_del, fall_del:real:= 0.0;
signal AND2_RISE_DELAY,AND2_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(AND2_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          AND2_RISE_DELAY,AND2_FALL_DELAY);


OUTPUT <= '1' after AND2_RISE_DELAY when (INPUT1 = '1' and INPUT2 = '1')
   else '0' after AND2_FALL_DELAY;


end BEHAVIORAL ;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;

-------------------------------
-- device AND3
-- file and3.vhd
-- 3 Input AND Gate
-------------------------------

entity AND3 is

 generic (DELAY:time;
--        LOAD:real;
        FANOUT:integer;
        VDD:real;
        TEMPERATURE:real;
        AND3_TABLE:cell_values);

 port (INPUT1 : in bit;
      INPUT2 : in bit;
      INPUT3 : in bit;
      OUTPUT : out bit);

end AND3;


architecture BEHAVIORAL of AND3 is


signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:real:= 0.0;
signal AND3_RISE_DELAY,AND3_FALL_DELAY:time:= 0 fs;


begin

GET_CELL_DATA(AND3_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          AND3_RISE_DELAY,AND3_FALL_DELAY);

OUTPUT <= '1' after AND3_RISE_DELAY when
     (INPUT1 = '1' and INPUT2 = '1' and INPUT3 = '1')
    else '0' after AND3_FALL_DELAY;


end BEHAVIORAL ;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-----------------------------
-- device BUFF
-- file buff.vhd
-- BUFF
-----------------------------

entity BUFF is

  generic (DELAY:      time;
--        LOAD:       real;
          FANOUT:     integer;
          VDD: real;
          TEMPERATURE: real;
          BUFF_TABLE:   cell_values);

  port (INPUT: in  bit;
        OUTPUT: out bit);

end BUFF;

-----------------------------------------------------------------------
architecture BEHAVIORAL of BUFF is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:  real:= 0.0;
signal BUFF_RISE_DELAY,BUFF_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(BUFF_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,FANOUT,
          cell_version,
          rise_del,fall_del,
          BUFF_RISE_DELAY,BUFF_FALL_DELAY);

process
begin


if (INPUT = '0')then
   OUTPUT <= INPUT after BUFF_FALL_DELAY;
else
   OUTPUT <= INPUT after BUFF_RISE_DELAY;
end if;

wait on INPUT;

end process;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-----------------------------
-- device  CMOS_IN
-- file  cmos_in.vhd
-- CMOS_IN
-----------------------------

entity CMOS_IN is

  generic (DELAY:       time;
--          LCAD:       real;
            FANOUT:     integer;
            VDD: real;
           TEMPERATURE: real;
            CMOS_IN_TABLE:   cell_values);

  port (INPUT: in  bit;
        OUTPUT: out bit);

end CMOS_IN;

-----------------------------------------------------------------------
architecture BEHAVIORAL of CMOS_IN is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:  real:= 0.0;
signal CMOS_IN_RISE_DELAY,CMOS_IN_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(CMOS_IN_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,FANOUT,
          cell_version,
          rise_del,fall_del,
          CMOS_IN_RISE_DELAY,CMOS_IN_FALL_DELAY);

process
begin


if (INPUT = '0')then
   OUTPUT <= INPUT after CMOS_IN_FALL_DELAY;
else
   OUTPUT <= INPUT after CMOS_IN_RISE_DELAY;
end if;

wait on INPUT;

end process;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-----------------------------
-- device CMOS_5VOUT
-- file cmos_5vout.vhd
-- CMOS_5VOUT
-----------------------------
entity CMOS_5VOUT is

  generic (DELAY:       time;
--        LOAD:        real;
          FANOUT:      integer;
          VDD: real;
          TEMPERATURE: real;
          CMOS_5VOUT_TABLE:   cell_values);

  port (INPUT: in  bit;
        EN: in  bit;
        OUTPUT: out bit);

end CMOS_5VOUT;
-----------------------------------------------------------------------
architecture BEHAVIORAL of CMOS_5VOUT is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:  real:= 0.0;
signal CMOS_5VOUT_RISE_DELAY,CMOS_5VOUT_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(CMOS_5VOUT_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,FANOUT,
          cell_version,
          rise_del,fall_del,
          CMOS_5VOUT_RISE_DELAY,CMOS_5VOUT_FALL_DELAY);

process
begin

if (EN = '1') then
  if (INPUT = '0') then
    OUTPUT <= '0' after CMOS_5VOUT_FALL_DELAY;
  else
    OUTPUT <= '1' after CMOS_5VOUT_RISE_DELAY;
  end if;
else
    OUTPUT <= '0' after CMOS_5VOUT_FALL_DELAY;
end if;
wait on INPUT;

end process;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-----------------------------
-- device CMOS_PAD
-- file cmos_pad.vhd
-- CMOS_PAD
-----------------------------

entity CMOS_PAD is

  generic (DELAY:      time;
--        LOAD:       real;
          FANOUT:     integer;
          VDD: real;
          TEMPERATURE: real;
          CMOS_PAD_TABLE:   cell_values);

  port (INPUT: in  bit;
        OUTPUT: out bit);

end CMOS_PAD;

-----------------------------------------------------------------
architecture BEHAVIORAL of CMOS_PAD is


signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:  real:= 0.0;
signal CMOS_PAD_RISE_DELAY,CMOS_PAD_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(CMOS_PAD_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,FANOUT,
          cell_version,
          rise_del,fall_del,
          CMOS_PAD_RISE_DELAY,CMOS_PAD_FALL_DELAY);

process
begin


if (INPUT = '0')then
   OUTPUT <= INPUT after CMOS_PAD_FALL_DELAY;
else
   OUTPUT <= INPUT after CMOS_PAD_RISE_DELAY;
end if;

wait on INPUT;

end process;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-----------------------
-- device EXOR
-- file exor.vhd
-- Exclusive OR gate
-----------------------
entity EXOR is

 generic (DELAY:time;
--        LOAD:real;
          FANOUT:integer;
          VDD:real;
          TEMPERATURE:real;
          EXOR_TABLE:cell_values);


 port (INPUT1: in  bit;
       INPUT2: in  bit;
       OUTPUT: out bit);

end EXOR;

architecture BEHAVIORAL of EXOR is

signal cell_version: String(1 to 16):="              ";
signal rise_del, fall_del:real:= 0.0;
signal EXOR_RISE_DELAY,EXOR_FALL_DELAY:time:= 0 fs;


begin

GET_CELL_DATA(EXOR_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          EXOR_RISE_DELAY,EXOR_FALL_DELAY);


OUTPUT <= '1' after EXOR_RISE_DELAY when
        ((INPUT1 = '1' and INPUT2 = '0') or (INPUT1 = '0' and INPUT2 = '1'))
    else '0' after EXOR_FALL_DELAY;


end BEHAVIORAL ;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-----------------------
-- device EXNOR
-- file exnor.vhd
-- Exclusive OR gate
-----------------------
entity EXNOR is

 generic (DELAY:time;
--        LOAD:real;
          FANOUT:integer;
          VDD:real;
          TEMPERATURE:real;
          EXNOR_TABLE:cell_values);


 port (INPUT1: in  bit;
       INPUT2: in  bit;
       OUTPUT: out bit);

end EXNOR;

architecture BEHAVIORAL of EXNOR is

signal cell_version: String(1 to 16):="              ";
signal rise_del, fall_del:real:= 0.0;
signal EXNOR_RISE_DELAY,EXNOR_FALL_DELAY:time:= 0 fs;


begin

GET_CELL_DATA(EXNOR_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          EXNOR_RISE_DELAY,EXNOR_FALL_DELAY);


OUTPUT <= '0' after EXNOR_FALL_DELAY when
        ((INPUT1 = '1' and INPUT2 = '0') or (INPUT1 = '0' and INPUT2 = '1'))
    else '1' after EXNOR_RISE_DELAY;


end BEHAVIORAL ;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-----------------------------
-- device INVERTER
-- file inverter.vhd
-- INVERTER
-----------------------------

entity INV is

  generic (DELAY:       time;
--   LOAD:       real;
   FANOUT:     integer;
        VDD: real;
        TEMPERATURE: real;
        INV_TABLE:   cell_values);

  port (INPUT: in  bit := '0';
      OUTPUT: out bit := '1');

end INV;

-----------------------------------------------------------------
architecture BEHAVIORAL of INV is


signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:  real:= 0.0;
signal INV_RISE_DELAY,INV_FALL_DELAY:  time:= 1 fs;

begin

GET_CELL_DATA(INV_TABLE,DELAY,
         LOAD,VDD,TEMPERATURE,FANOUT,
         cell_version,
         rise_del,fall_del,
         INV_RISE_DELAY,INV_FALL_DELAY);

process
begin


if (INPUT = '0')then
   OUTPUT <= not INPUT after INV_RISE_DELAY;
else
   OUTPUT <= not INPUT after INV_FALL_DELAY;
end if;

wait on INPUT;

end process;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-----------------------------
-- device HD_BUFF
-- file hd_buff.vhd
-- HD_BUFF
-----------------------------

entity HD_BUFF is

  generic (DELAY:       time;
--        LOAD:        real;
          FANOUT:      integer;
          VDD: real;
          TEMPERATURE: real;
          HD_BUFF_TABLE:   cell_values);

  port (INPUT: in  bit;
        OUTPUT: out bit);

end HD_BUFF;

-----------------------------------------------------------------
architecture BEHAVIORAL of HD_BUFF is


signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:  real:= 0.0;
signal HD_BUFF_RISE_DELAY,HD_BUFF_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(HD_BUFF_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,FANOUT,
          cell_version,
          rise_del,fall_del,
          HD_BUFF_RISE_DELAY,HD_BUFF_FALL_DELAY);

process
begin


if (INPUT = '0')then
   OUTPUT <= INPUT after HD_BUFF_FALL_DELAY;
else
   OUTPUT <= INPUT after HD_BUFF_RISE_DELAY;
end if;

wait on INPUT;

end process;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;

-----------------------------
-- device NAND2
-- file nand2.vhd
-- 2 Input NAND Gate
-----------------------------

entity NAND2 is

 generic (DELAY:  time;
--        LOAD:  real;
          FANOUT:  integer;
          INITDELAY:  time;
          VDD:  real;
          TEMPERATURE:  real;
          NAND2_TABLE:  cell_values);

 port (INPUT1 : in bit;
       INPUT2 : in bit;
       OUTPUT : out bit:= '0');

end NAND2;

-----------------------------------------------------------------
-- INITDELAY is a finite delay that is added to the nand2
-- gate when this gate is used to build a latch. It is
-- necessary to offset the delay of one gate by 1 fs in the
-- latch so that the simulation can initialize itself at time
-- zero.
-----------------------------------------------------------------

architecture BEHAVIORAL of NAND2 is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:real:= 0.0;
signal NAND2_RISE_DELAY,NAND2_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(NAND2_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          NAND2_RISE_DELAY,NAND2_FALL_DELAY);

OUTPUT <= '0' after NAND2_FALL_DELAY + INITDELAY when
    (INPUT1 = '1' and INPUT2 = '1')
   else '1' after NAND2_RISE_DELAY + INITDELAY ;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;

-------------------------------
-- device NAND3
-- file nand3.vhd
-- 2 Input NAND Gate
-------------------------------

entity NAND3 is

 generic (DELAY:  time;
--        LOAD:  real;
          FANOUT:  integer;
          INITDELAY:  time;
          VDD:  real;
          TEMPERATURE:  real;
          NAND3_TABLE:  cell_values);

 port (INPUT1 : in bit;
       INPUT2 : in bit;
       INPUT3 : in bit;
       OUTPUT : out bit:= '0');

end NAND3;
-------------------------------------------------------------------------
-- INITDELAY is a finite delay that is added to the nand3
-- gate when this gate is used to build a latch. It is
-- necessary to offset the delay of one gate by 1fs in the
-- latch so that the simulation can initialize itself at time
-- zero.
-------------------------------------------------------------------------

architecture BEHAVIORAL of NAND3 is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:real:= 0.0;
signal NAND3_RISE_DELAY,NAND3_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(NAND3_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          NAND3_RISE_DELAY,NAND3_FALL_DELAY);

OUTPUT <= '0' after NAND3_FALL_DELAY + INITDELAY when
          (INPUT1 = '1' and INPUT2 = '1' and INPUT3 = '1')
          else '1' after NAND3_RISE_DELAY + INITDELAY ;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;

-------------------------------
-- device NAND4
-- file nand4.vhd
-- 2 Input NAND Gate
-------------------------------

entity NAND4 is

 generic (DELAY:  time;
--        LOAD:  real;
          FANOUT:  integer;
          INITDELAY:  time;
          VDD:  real;
          TEMPERATURE:  real;
          NAND4_TABLE:  cell_values);

 port (INPUT1 : in bit;
       INPUT2 : in bit;
       INPUT3 : in bit;
       INPUT4 : in bit;
       OUTPUT : out bit);

end NAND4;
-------------------------------------------------------------------------
-- INITDELAY is a finite delay that is added to the nand4
-- gate when this gate is used to build a latch. It is
-- necessary to offset the delay of one gate by 1fs in the
-- latch so that the simulation can initialize itself at time
-- zero.
-------------------------------------------------------------------------
architecture BEHAVIORAL of NAND4 is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:real:= 0.0;
signal NAND4_RISE_DELAY,NAND4_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(NAND4_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          NAND4_RISE_DELAY,NAND4_FALL_DELAY);

OUTPUT <= '0' after NAND4_FALL_DELAY + INITDELAY when
          (INPUT1 = '1' and INPUT2 = '1' and INPUT3 = '1' and INPUT4 = '1'
   else '1' after NAND4_RISE_DELAY + INITDELAY ;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;

-----------------------------
-- device NOR2
-- file nor2.vhd
-- 2 Input NOR2 Gate
-----------------------------

entity NOR2 is

 generic (DELAY:  time;
--        LOAD:  real;
          FANOUT:  integer;
          INITDELAY:  time;
          VDD:  real;
          TEMPERATURE:  real;
          NOR2_TABLE:  cell_values);

 port (INPUT1 : in bit;
       INPUT2 : in bit;
       OUTPUT : out bit:= '0');

end NOR2;

-----------------------------------------------------------------
-- INITDELAY is a finite delay that is added to the nor2
-- gate when this gate is used to build a latch. It is
-- necessary to offset the delay of one gate by 1fs in the
-- latch so that the simulation can initialize itself at time
-- zero.
-----------------------------------------------------------------

architecture BEHAVIORAL of NOR2 is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:real:= 0.0;
signal NOR2_RISE_DELAY,NOR2_FALL_DELAY: time:= 0 fs;

begin

GET_CELL_DATA(NOR2_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          NOR2_RISE_DELAY,NOR2_FALL_DELAY);

OUTPUT <= '1' after NOR2_RISE_DELAY + INITDELAY when
    (INPUT1 = '0' and INPUT2 = '0')
   else '0' after NOR2_FALL_DELAY + INITDELAY;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;

-------------------------------
-- device NOR3
-- file nor3.vhd
-- 3 Input NOR Gate
-------------------------------

entity NOR3 is

 generic (DELAY:  time;
--        LOAD:  real;
          FANOUT:  integer;
          INITDELAY:  time;
          VDD:  real;
          TEMPERATURE:  real;
          NOR3_TABLE:  cell_values);

 port (INPUT1 : in bit;
       INPUT2 : in bit;
       INPUT3 : in bit;
       OUTPUT : out bit);

end NOR3;
-----------------------------------------------------------------
-- INITDELAY is a finite delay that is added to the nor3
-- gate when this gate is used to build a latch. It is
-- necessary to offset the delay of one gate by 1fs in the
-- latch so that the simulation can initialize itself at time
-- zero.
-----------------------------------------------------------------

architecture BEHAVIORAL of NOR3 is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:real:= 0.0;
signal NOR3_RISE_DELAY,NOR3_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(NOR3_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,
          FANOUT,
          cell_version,
          rise_del,fall_del,
          NOR3_RISE_DELAY,NOR3_FALL_DELAY);

OUTPUT <= '1' after NOR3_RISE_DELAY + INITDELAY when
          (INPUT1 = '0' and INPUT2 = '0' and INPUT3 = '0')
    else '0' after NOR3_FALL_DELAY + INITDELAY;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;

-------------------------------
-- device NOR4
-- file nor4.vhd
-- 4 Input NOR Gate
-------------------------------

entity NOR4 is

 generic (DELAY:  time;
--        LOAD:  real;
        FANOUT:  integer;
  INITDELAY:  time;
        VDD:  real;
        TEMPERATURE:  real;
        NOR4_TABLE:  cell_values);

 port (INPUT1 : in bit;
       INPUT2 : in bit;
       INPUT3 : in bit;
       INPUT4 : in bit;
       OUTPUT : out bit);

end NOR4;
-------------------------------------------------------------------------
-- INITDELAY is a finite delay that is added to the nor4
-- gate when this gate is used to build a latch. It is
-- necessary to offset the delay of one gate by 1fs in the
-- latch so that the simulation can initialize itself at time
-- zero.
-------------------------------------------------------------------------
architecture BEHAVIORAL of NOR4 is

signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:real:= 0.0;
signal NOR4_RISE_DELAY,NOR4_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(NOR4_TABLE,DELAY,
         LOAD,VDD,TEMPERATURE,
         FANOUT,
         cell_version,
         rise_del,fall_del,
         NOR4_RISE_DELAY,NOR4_FALL_DELAY);

OUTPUT <= '1' after NOR4_RISE_DELAY + INITDELAY when
         (INPUT1 = '0' and INPUT2 = '0' and INPUT3 = '0' and INPUT4 = '0')
        else '0' after NOR4_FALL_DELAY + INITDELAY;
end BEHAVIORAL;
```

```
use work.GEM_CONSTANTS.all;
use work.GEM_DELAYS.all;
use work.tables.all;
-- use STD.Simulator_Standard.all;
-------------------------------
-- device OUT_BUFF
-- file out_buff.vhd
-- OUT_BUFF
-------------------------------

entity OUT_BUFF is

  generic (DELAY:      time;
--        LOAD:      real;
          FANOUT:    integer;
          VDD: real;
          TEMPERATURE: real;
          OUT_BUFF_TABLE:   cell_values);

  port (INPUT: in  bit;
        OUTPUT: out bit);

end OUT_BUFF;

-------------------------------------------------------------------------

architecture BEHAVIORAL of OUT_BUFF is


signal cell_version: String(1 to 16):="              ";
signal rise_del,fall_del:  real:= 0.0;
signal OUT_BUFF_RISE_DELAY,OUT_BUFF_FALL_DELAY:time:= 0 fs;

begin

GET_CELL_DATA(OUT_BUFF_TABLE,DELAY,
          LOAD,VDD,TEMPERATURE,FANOUT,
          cell_version,
          rise_del,fall_del,
          OUT_BUFF_RISE_DELAY,OUT_BUFF_FALL_DELAY);

process
begin

if (INPUT = '0')then
   OUTPUT <= INPUT after OUT_BUFF_FALL_DELAY;
else
   OUTPUT <= INPUT after OUT_BUFF_RISE_DELAY;
end if;

wait on INPUT;

end process;
end BEHAVIORAL;
```

```
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-----------------------------------------
-- device DFF_PEDG_RBAR
-- file dff_pedg_rbar.vhd
-- D Flip-Flop with reset
-----------------------------------------

entity dff_pedg_rbar is

generic (Q_FANOUT: integer;
         QN_FANOUT: integer;
         VDD: real;
         TEMPERATURE: real);

port (D: in bit;
      CLK: in bit;
      RESET: in bit;
      Q: out bit;
      QBAR: out bit);

end dff_pedg_rbar;

-----------------------------------------------------------
-- Structural architecture for the D-Flip-Flop with Reset.
-- This D-Flip-Flop is a positive edge triggered flip-flop.
-- The Q output shall go to a logic "1" upon the rising edge
-- of the CLK input if the D input is a logic "1" and the
-- RESET input is a logic "1".
-- The Q output shall go to a logic "0" upon the rising edge
-- of the CLK input if the D input is a logic "0".
-- The QN output shall always be the complement of the Q output.
-- A logic "0" on the RESET input shall force the Q output to a
-- logic "0" and the QN output to a logic "1" regardless of the
-- state of the CLK input.
-----------------------------------------------------------

architecture struct of dff_pedg_rbar is

   component buff
      generic (DELAY:time;FANOUT:integer;VDD:real;
               TEMPERATURE:real;
               BUFF_TABLE:cell_values:=work.tables.buff_table);
      port (INPUT: in bit;
            OUTPUT: out bit);
   end component;
```

```
  component nand2
      generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
            VDD:real;TEMPERATURE:real;
            NAND2_TABLE:cell_values:=work.tables.nand2_table);
      port (INPUT1: in  bit;
          INPUT2: in  bit;
          OUTPUT: out bit);
  end component;

  component nand3
      generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
            VDD:real;TEMPERATURE:real;
            NAND3_TABLE:cell_values:=work.tables.nand3_table);
      port (INPUT1: in  bit;
          INPUT2: in  bit;
          INPUT3: in  bit;
          OUTPUT: out bit);
  end component;

  signal T3, T2, T1, T0: bit;
  signal T4, T5:       bit;

  for all:buff use entity work.buff(behavioral);
  for all:nand2 use entity work.nand2(behavioral);
  for all:nand3 use entity work.nand3(behavioral);

begin


  A0: nand2
      generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
      port map (T3, T1, T0);
  A1: nand3
      generic map (DELAY=> 1 ns, FANOUT=>3, INITDELAY=> 1 fs,
                VDD=>VDD, TEMPERATURE=>TEMPERATURE)
      port map (T0, RESET, CLK, T1);
  A2: nand3
      generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
      port map (T1, CLK, T3, T2);
  A3: nand3
      generic map (DELAY=> 1 ns, FANOUT=>2, INITDELAY=> 1 fs,
                VDD=>VDD, TEMPERATURE=>TEMPERATURE)
      port map (T2, D, RESET, T3);
  A4: nand2
      generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
      port map (T1, T5, T4);
  A5: nand3
      generic map (DELAY=> 1 ns, FANOUT=>2, INITDELAY=> 1 fs,
                VDD=>VDD, TEMPERATURE=>TEMPERATURE)
      port map (T4, T2, RESET, T5);
```

```
  B1: buff
      generic map (DELAY=> 1 ns, FANOUT=>Q_FANOUT, VDD=>VDD,
                   TEMPERATURE=>TEMPERATURE)
      port map (T4, Q);

  B2: buff
      generic map (DELAY=> 1 ns, FANOUT=>QN_FANOUT, VDD=>VDD,
                   TEMPERATURE=>TEMPERATURE)
      port map (T5, QBAR);

end struct;

use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-------------------------------------
-- device DFF_PEDG_SRBAR
-- file dff_pedg_srbar
-- D Flip-Flop with preset and clear
-------------------------------------

entity dff_pedg_srbar is

generic (Q_FANOUT:integer;
 QN_FANOUT:integer;
 VDD:real;
 TEMPERATURE:real);

port (D:inbit;
    CLK:inbit;
    RESET:inbit;
    SET:inbit;
    Q:outbit;
    QBAR:outbit);

end dff_pedg_srbar;

-------------------------------------------------------
-- Structural architecture for the D-Flip-Flop with Reset.
-- and Preset capabiliaity. This D-Flip-Flop is a positive edge
-- triggered flip-flop. The Q output shall go to a logic "1"
-- upon the rising edge of the CLK input if the D input is a
-- logic "1" and the RESET input is a logic "1". and the PRESET
-- input is a logic "1". The Q output shall go to a logic "0" upon
-- the rising edge of the CLK input if the D input is a logic "0".
-- The QN output shall always be the complement of the Q output.
-- A logic "0" on the RESET input shall force the Q output to a
-- logic "0" and the QN output to a logic "1" regardless of the
-- state of the CLK input.
-- A logic "0" on the PRESET input shall force the Q output to a
-- logic "1" and the QN output to a logic "0" regardless of the
-- state of the CLK input.
-------------------------------------------------------
```

```
architecture struct of dff_pedg_srber is

  component buff
      generic (DELAY:time;FANOUT:integer;VDD:real;
               TEMPERATURE:real;
               BUFF_TABLE:cell_values:=work.tables.buff_table);
      port (INPUT: in bit;
            OUTPUT: out bit);
  end component;

  component nand3
      generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
               VDD:real;TEMPERATURE:real;
               NAND3_TABLE:cell_values:=work.tables.nand3_table);
      port (INPUT1: in bit;
            INPUT2: in bit;
            INPUT3: in bit;
            OUTPUT: out bit);
  end component;


  signal T3, T2, T1, T0: bit;
  signal T4, T5:       bit;

  for all:buff use entity work.buff(behavioral);
  for all:nand3 use entity work.nand3(behavioral);

begin

  A0: nand3
      generic map (DELAY=>  1 ns, FANOUT=>1, VDD=>VDD,
                   TEMPERATURE=>TEMPERATURE)
      port map (SET, T3, T1, T0);

  A1: nand3
      generic map (DELAY=>  1 ns, FANOUT=>3, INITDELAY=> 1 fs,
                   VDD=>VDD, TEMPERATURE=>TEMPERATURE)
      port map (T0, RESET, CLK, T1);

  A2: nand3
      generic map (DELAY=>  1 ns, FANOUT=>2, VDD=>VDD,
                   TEMPERATURE=>TEMPERATURE)
      port map (T1, CLK, T3, T2);

  A3: nand3
      generic map (DELAY=>  1 ns. FANOUT=>2, INITDELAY=> 1 fs,
                   VDD=>VDD, TEMPERATURE=>TEMPERATURE)
      port map (T2, D, RESET, T3);

  A4: nand3
      generic map (DELAY=>  1 ns, FANOUT=>2, VDD=>VDD,
                   TEMPERATURE=>TEMPERATURE)
      port map (SET, T1, T5, T4);
```

```
   A5: nand3
       generic map (DELAY=> 1 ns, FANOUT=>2, INITDELAY=> 1 fs,
                    VDD=>VDD, TEMPERATURE=>TEMPERATURE)
       port map (T4, T2, RESET, T5);

   B1: buff
       generic map (DELAY=> 1 ns, FANOUT=>Q_FANOUT, VDD=>VDD,
                    TEMPERATURE=>TEMPERATURE)
       port map (T4, Q);

   B2: buff
       generic map (DELAY=> 1 ns, FANOUT=>QN_FANOUT, VDD=>VDD,
                    TEMPERATURE=>TEMPERATURE)
       port map (T5, QBAR);

end struct;

use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-------------------------------------
-- device AOR
-- file aor.vhd
-- AND/OR function
-------------------------------------

entity aor is

generic (A0_FANOUT:integer;
 B0_FANOUT:integer;
 C0_FANOUT:integer;
 D0_FANOUT:integer;
 VDD:real;
 TEMPERATURE:real);

port (A1, A2:inbit;
    B1, B2:inbit;
    C1, C2:inbit;
    D1, D2:inbit;
    K1, K2:inbit;
    A0, B0, C0, D0:outbit);

end aor;

-----------------------------------------------------------
-- Structural architecture for the AND/OR (AOR) function.
-- THE outputs A0, A1, A2, A3 shall be forced to a logic "1"
-- if either of inputs, K1 or K2, are a logic "1", Or if:
-- A logic "1" on either A1 or A2 forces A0 to a logic "1".
-- A logic "1" on either B1 or B2 forces B0 to a logic "1".
-- A logic "1" on either C1 or C2 forces C0 to a logic "1".
-- A logic "1" on either D1 or D2 forces D0 to a logic "1".
-- Else the outputs A0, A1, A2, A3 are a logic "0".
```

```
---------------------------------------------------------

architecture struct of aor is

  component nand2
generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
 VDD:real; TEMPERATURE:real;
      NAND2_TABLE:cell_values:=work.tables.r.and2_table);
port (INPUT1:in  bit;
    INPUT2:in  bit;
    OUTPUT:out bit);
  end component;

  signalI16_O, I17_O:bit;
  signalI18_O, I19_O:bit;
  signalI22_O, I23_O:bit;
  signalI24_O, I25_O:bit;

  for all:nand2 use entity work.nand2(behavioral);

begin


  I16: nand2
       generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
       port map (A1, K1, I16_O);

  I17: nand2
       generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
       port map (A2, K2, I17_O);

  I18: nand2
       generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
       port map (B1, K1, I18_O);

  I19: nand2
       generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
       port map (B2, K2, I19_O);

  I25: nand2
       generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
       port map (C1, K1, I25_O);

  I24: nand2
       generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
       port map (C2, K2, I24_O);

  I23: nand2
```

```
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (D1, K1, I23_O);

  I22: nand2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (D2, K2, I22_O);

  I20: nand2
        generic map (DELAY=>1 ns, FANOUT=>A0_FANOUT, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (I16_O, I17_O, A0);

  I21: nand2
        generic map (DELAY=>1 ns, FANOUT=>B0_FANOUT, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (I18_O, I19_O, B0);

  I27: nand2
        generic map (DELAY=>1 ns, FANOUT=>C0_FANOUT, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (I25_O, I24_O, C0);

  I26: nand2
        generic map (DELAY=>1 ns, FANOUT=>D0_FANOUT, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (I23_O, I22_O, D0);

end struct;



use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-------------------------------------
-- device SYN4CTR
-- file syn4ctr.vhd
-- Synchronous four bit up/down counter with
-- reset and carry output
-------------------------------------

entity syn4ctr is

    generic (A_FANOUT:integer;
             B_FANOUT:integer;
             C_FANOUT:integer;
             D_FANOUT:integer;
             CO_FANOUT:integer;
             VDD:real;
             TEMPERATURE:real);
```

```
    port (CLK, UP, RST, CI:inbit;
          A:inoutbit;
          B, C, D, CO:outbit);

end syn4ctr;

-------------------------------------------------------
-- Structural architecture for the 4-bit synchronous up-down
-- counter with asynchronous reset, count enable and carry out
-- capability. A logic "1" on the UP input enables count up
-- mode and a logic "0" on the UP input enables count down.
-- The counter shall increment or decrement upon the rising
-- edge of the CLK input providing the RESET input is a logic "0"
-- and the CI input is a logic "1". The CO output shall go to a
-- logic "1" upon full count, outputs = "1111" and CI = "1" or
-- outputs = "0000" and CI = "0". The CI input and the CO output
-- allows cascading of multiple counters into larger counters.
-- When the RESET input is a logic "1" the counters shall reset
-- regardless of the CLK input.
-- Reset in count up mode is outputs = "0000".
-- Reset in count down mode is outputs = "1111".
-- A is the LSB output and D is the MSB output.
-------------------------------------------------------
architecture struct of syn4ctr is

  component inv
    generic (DELAY:time;FANOUT:integer;VDD:real;
             TEMPERATURE:real;
             INV_TARLE:cell_values:=work.tables.inv_table);
    port (INPUT:in bit;
          OUTPUT:out bit);
  end component;

  component and2
    generic (DELAY:time;FANOUT:integer;VDD:real;
             TEMPERATURE:real;
             AND2_TABLE:cell_values:=work.tables.and2_table);
    port (INPUT1:in bit;
          INPUT2:in bit;
          OUTPUT:out bit);
  end component;

  component nand2
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
             VDD:real;TEMPERATURE:real;
             NAND2_TABLE:cell_values:=work.tables.nand2_table);
    port (INPUT1:in bit;
          INPUT2:in bit;
          OUTPUT:out bit);
  end component;

  component nand3
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
             VDD:real;TEMPERATURE:real;
```

```
        NAND3_TABLE:cell_values:=work.tables.nand3_table);
    port (INPUT1:in bit;
          INPUT2:in bit;
          INPUT3:in bit;
          OUTPUT:out bit);
end component;

component nor2
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
             VDD:real;TEMPERATURE:real;
             NOR2_TABLE:cell_values:=work.tables.nor2_table);
    port (INPUT1:in bit;
          INPUT2:in bit;
          OUTPUT:out bit);
end component;

component nor3
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
             VDD:real;TEMPERATURE:real;
             NOR3_TABLE:cell_values:=work.tables.nor3_table);
    port (INPUT1:in bit;
          INPUT2:in bit;
          INPUT3:in bit;
          OUTPUT:out bit);
end component;

component nor4
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
             VDD:real;TEMPERATURE:real;
             NOR4_TABLE:cell_values:=work.tables.nor4_table);
    port (INPUT1:in bit;
          INPUT2:in bit;
          INPUT3:in bit;
          INPUT4:in bit;
          OUTPUT:out bit);
end component;

component exnor
    generic (DELAY:time;FANOUT:integer;VDD:real;
             TEMPERATURE:real;
             EXNOR_TABLE:cell_values:=work.tables.exnor_table);
    port (INPUT1:in bit;
          INPUT2:in bit;
          OUTPUT:out bit);
end component;

component dff_pedg_rbar
    generic (Q_FANOUT:integer;QN_FANOUT:integer;
             VDD:real;TEMPERATURE:real);
    port (D:in bit;
          CLK:inbit;
          RESET:inbit;
          Q:outbit;
          QBAR:outbit);
```

```
  end component;

  signal CLKN, RSTN, UPN:bit;
  signal QAN, QBN, QCN, QDN:bit;
  signal I206_O, I207_O, I208_O, I209_O:bit;
  signal I210_O, I211_O, I212_O, I214_O:bit;
  signal I215_O, I216_O, I217_O, I218_O:bit;
  signal I225_O, I226_O, I227_O, I228_O:bit;
  signal I201_O, I281_O, I187_O, I185_O:bit;
  signal I229_O, I230_O, I200_O, I199_O:bit;
  signal I196_O, I197_O, I198_O, I222_O:bit;
  signal I302_O, I183_O, I223_O:      bit;

  for all:inv use entity work.inv(behavioral);
  for all:and2 use entity work.and2(behavioral);
  for all:nand2 use entity work.nand2(behavioral);
  for all:nand3 use entity work.nand3(behavioral);
  for all:nor2 use entity work.nor2(behavioral);
  for all:nor3 use entity work.nor3(behavioral);
  for all:nor4 use entity work.nor4(behavioral);
  for all:exnor use entity work.exnor(behavioral);
  for all:dff_pedg_rbar use entity work.dff_pedg_rbar(struct);

begin


  I220: inv
        generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (CLK, CLKN);

  I219: inv
        generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (RST, RSTN);

  I221: inv
        generic map (DELAY=>1 ns, FANOUT=>6, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (UP, UPN);

  I222: inv
        generic map (DELAY=>1 ns, FANOUT=>6, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (UPN, I222_O);

  I225: exnor
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (QAN, CI, I225_O);

  I23:  dff_pedg_rbar
        generic map (Q_FANOUT=>A_FANOUT, QN_FANOUT=>8, VDD=>VDD,
           TEMPERATURE=>TEMPERATURE)
```

```
        port map (I225_O, CLKN, RSTN, A, QAN);

I217: and2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (A, UPN, I217_O);

I218: and2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I222_O, QAN, I218_O);

I223: nor2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I217_O, I218_O, I223_O);

I187: and2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I223_O, CI, I187_O);

I226: exnor
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I187_O, QBN, I226_O);

I1:  dff_pedg_rbar
        generic map (Q_FANOUT=>B_FANOUT, QN_FANOUT=>7, VDD=>VDD,
           TEMPERATURE=>TEMPERATURE)
        port map (I226_O, CLKN, RSTN, B, QBN);

I281: nand2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (QBN, QAN, I281_O);

I212: and2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (UPN,I281_O, I212_O);

I211: and2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I222_O, QBN, I211_O);

I210: and2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I222_O, QAN, I210_O);

I228: nor3
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
```

```
        TEMPERATURE=>TEMPERATURE)
    port map (I212_O, I211_O, I210_O, I228_O);

I185: and2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (I228_O, CI, I185_O);

I214: exnor
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (I185_O, QCN, I214_O);

I9:  dff_pedg_rbar
    generic map (Q_FANOUT=>C_FANOUT, QN_FANOUT=>5, VDD=>VDD,
      TEMPERATURE=>TEMPERATURE)
    port map (I214_O, CLKN, RSTN, C, QCN);

I201: nand3
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (QCN, QBN, QAN, I201_O);

I206: and2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (UPN, I201_O, I206_O);

I207: and2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (I222_O, QCN, I207_O);

I208: and2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (I222_O, QBN, I208_O);

I209: and2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (I222_O, QAN, I209_O);

I227: nor4
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (I206_O, I207_O, I208_O, I209_O, I227_O);

I183: and2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
    port map (I227_O, CI, I183_O);

I215: exnor
```

```
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I183_O, QDN, I215_O);

    I12: dff_pedg_rbar
        generic map (Q_FANOUT=>D_FANOUT, QN_FANOUT=>3, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
        port map (I215_O, CLKN, RSTN, D, QDN);

    I216: inv
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (CI, I216_O);

    I230: nor3
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I216_O, QDN, QCN, I230_O);

    I229: nor3
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (QBN, QAN, UPN, I229_O);

    I196: nand2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I230_O, I229_O, I196_O);

    I200: nand3
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (QDN, QCN, QBN, I200_O);

    I199: nand3
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (QAN, UPN, CI, I199_O);

    I198: nor2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I200_O, I199_O, I198_O);

    I197: inv
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I198_O, I197_O);

    I302: nand2
        generic map (DELAY=>1 ns, FANOUT=>CO_FANOUT, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I196_O, I197_O, CO);
end struct;
```

```
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-------------------------------------
-- device UP5CTR
-- file up5ctr.vhd
-- Five bit up counter with reset
-------------------------------------

entity up5ctr is

generic (A_FANOUT:integer;
 B_FANOUT:integer;
 C_FANOUT:integer;
 D_FANOUT:integer;
 E_FANOUT:integer;
 VDD:real;
 TEMPERATURE:real);

port (CLK, RST:inbit;
    A, B, C, D, E:outbit);

end up5ctr;

-------------------------------------------------------
-- Structural architecture for a synchronous 5-bit counter with
-- asynchronous Reset. The counter shall increment by one upon
-- each rising edge of the CLK input if the RESET input is a
-- logic "0". The counter shall rollover to a count of "00000" if
-- the present count is "11111".
-- When the RESET input is a logic "1" the output shall be "00000"
-- regardless of the state of the CLK input.
-- A is the LSB-bit and E is the MSB-bit.
-------------------------------------------------------

architecture struct of up5ctr is

  component inv
      generic (DELAY:time;FANOUT:integer;VDD:real;
               TEMPERATURE:real;
               INV_TABLE:cell_values:=work.tables.inv_table);
      port (INPUT:in  bit;
            OUTPUT:out bit);
  end component;

  component nor2
      generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
               VDD:real;TEMPERATURE:real;
              NOR2_TABLE:cell_values:=work.tables.nor2_table);
      port (INPUT1:in  bit;
            INPUT2:in  bit;
            OUTPUT:out bit);
  end component;
```

```
component nor3
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
          VDD:real;TEMPERATURE:real;
          NOR3_TABLE:cell_values:=work.tables.nor3_table);
    port (INPUT1:in  bit;
          INPUT2:in  bit;
          INPUT3:in  bit;
          OU1PUT:out bit);
end component;

component nor4
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
          VDD:real;TEMPERATURE:real;
          NOR4_TABLE:cell_values:=work.tables.nor4_table);
    port (INPUT1:in  bit;
          INPUT2:in  bit;
          INPUT3:in  bit;
          INPUT4:in  bit;
          OUTPUT:out bit);
end component;

component exnor
    generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          EXNOR_TABLE:cell_values:=work.tables.exnor_table);
    port (INPUT1:in  bit;
          INPUT2:in  bit;
          OUTPUT:out bit);
end component;

component dff_pedg_rbar
    generic (Q_FANOUT:integer;QN_FANOUT:integer; VDD:real;
          TEMPERATURE:real);
    port (D:in bit;
          CLK:inbit;
          RESET:inbit;
          Q:outbit;
          QBAR:outbit);
end component;

signal CLKN, RSTN, QA, QAN:bit;
signal QBN, QCN, QDN, QEN:bit;
signal I106_O, I107_O, I108_O:bit;
signal I109_O, I110_O, I111_O,I112_O:bit;


for all:inv use entity work.inv(behavioral);
for all:nor2 use entity work.nor2(behavioral);
for all:nor3 use entity work.nor3(behavioral);
for all:nor4 use entity work.nor4(behavioral);
for all:exnor use entity work.exnor(behavioral);
for all:dff_pedg_rbar use entity work.dff_pedg_rbar(struct);
```

```
begin

  I114: inv
        generic map (DELAY=>1 ns, FANOUT=>5, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (CLK, CLKN);

  I124: inv
        generic map (DELAY=>1 ns, FANOUT=>5, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (RST, RSTN);

  I109: exnor
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (QA, QBN, I109_O);

  I110: nor2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (QBN, QAN, I110_O);

  I111: nor3
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (QCN, QBN, QAN, I111_O);

  I112: nor4
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (QDN, QCN, QBN, QAN, I112_O);

  I106: exnor
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (I112_O, QEN, I106_O);

  I107: exnor
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (I111_O, QDN, I107_O);

  I108: exnor
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
              TEMPERATURE=>TEMPERATURE)
        port map (I110_O, QCN, I108_O);

  I101: dff_pedg_rbar
        generic map (Q_FANOUT=>1, QN_FANOUT=>5, VDD=>VDD,
           TEMPERATURE=>TEMPERATURE)
        port map (QAN, CLKN, RSTN, QA, QAN);
```

```
  I33: inv
          generic map (DELAY=>1 ns, FANOUT=>A_FANOUT, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
          port map (QAN, A);

  I102: dff_pedg_rbar
          generic map (Q_FANOUT=>B_FANOUT, QN_FANOUT=>4, VDD=>VDD,
             TEMPERATURE=>TEMPERATURE)
          port map (I109_O, CLKN, RSTN, B, QBN);

  I103: dff_pedg_rbar
          generic map (Q_FANOUT=>C_FANOUT, QN_FANOUT=>3, VDD=>VDD,
             TEMPERATURE=>TEMPERATURE)
          port map (I108_O, CLKN, RSTN, C, QCN);

  I104: dff_pedg_rbar
          generic map (Q_FANOUT=>D_FANOUT, QN_FANOUT=>2, VDD=>VDD,
             TEMPERATURE=>TEMPERATURE)
          port map (I107_O, CLKN, RSTN, D, QDN);

  I105: dff_pedg_rbar
          generic map (Q_FANOUT=>E_FANOUT, QN_FANOUT=>1, VDD=>VDD,
             TEMPERATURE=>TEMPERATURE)
          port map (I106_O, CLKN, RSTN, E, QEN);

end struct;


use STD.Standard.all;
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-------------------------------------------------------------------------
-- This is the 9 stage ripple counter module which is used in the PRC-70  --
-- 395 chip model                                                         --
-------------------------------------------------------------------------

entity rctr_9stage is

   generic (q9_fanout:   integer;
        VDD:        real;
        TEMPERATURE:  real);

   port   (clk9, res9:  in   bit;
           q9:          out  bit);

end rctr_9stage;

architecture struct of rctr_9stage is

   component dff_pedg_rbar
      generic (Q_FANOUT: integer; QN_FANOUT:integer;
            VDD:real; TEMPERATURE:real);
```

```
    port (D:     in   bit;
          CLK:   in   bit;
          RESET: in   bit;
          Q:     out  bit;
          QBAR:  out  bit);

  end component;

  component hd_buff
    generic (DELAY:time;FANOUT:integer;VDD:real;
             TEMPERATURE:real;
             HD_BUFF_TABLE:cell_values:=work.tables.hd_buff_table);
    port (INPUT:   in   bit;
          OUTPUT:  out  bit);

  end component;

  component inv
    generic (DELAY:time;FANOUT:integer;VDD:real;
             TEMPERATURE:real;
             INV_TABLE:cell_values:=work.tables.inv_table);
    port (INPUT:   in   bit;
          OUTPUT:  out  bit);

  end component;

  signal VCC: bit:= '1';
  signal GND: bit:= '0';

  signal I901_Q, I901_QN:      bit;
  signal I902_Q, I902_QN:      bit;
  signal I903_Q, I903_QN:      bit;
  signal I904_N, I905_N:       bit;
  signal I906_Q, I906_QN:      bit;
  signal I907_Q, I907_QN:      bit;
  signal I908_Q, I908_QN:      bit;
  signal I909_Q, I909_QN:      bit;
  signal I910_Q, I910_QN:      bit;
  signal I911_Q, I911_QN:      bit;


  for all: dff_pedg_rbar use entity work.dff_pedg_rbar(struct);
  for all: hd_buff use entity work.hd_buff(behavioral);
  for all: inv use entity work.inv(behavioral);


begin

  I901: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                 TEMPERATURE=> TEMPERATURE)
    port map (I901_QN,clk9,I905_N,open,I901_QN);
```

```
I902: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I902_QN,I901_QN,I905_N,open,I902_QN);

I903: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I903_QN,I902_QN,I905_N,open,I903_QN);

I904: inv
    generic map (DELAY=> 1ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (res9, I904_N);

I905: hd_buff
    generic map (DELAY=> 1 ns, FANOUT=> 9, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I904_N, I905_N);

I906: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I906_QN,I903_QN,I905_N,open,I906_QN);

I907: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I907_QN,I906_QN,I905_N,open,I907_QN);

I908: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I908_QN,I907_QN,I905_N,open,I908_QN);

I909: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I909_QN,I908_QN,I905_N,open,I909_QN);

I910: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I910_QN,I909_QN,I905_N,open,I910_QN);

I911: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I911_QN,I910_QN,I905_N,q9,I911_QN);

end struct;
```

```
use STD.Standard.all;
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-----------------------------------------------------------------
-- This is the 13 stage ripple counter module used in the PRC-70 395 chip --
-----------------------------------------------------------------

entity rctr_13stage is

   generic (q13_fanout:  integer;
            VDD:         real;
            TEMPERATURE: real);

   port  (clk13, res13:  in   bit;
            q13:         out  bit);

end rctr_13stage;

architecture struct of rctr_13stage is

   component dff_pedg_rbar
      generic (Q_FANOUT: integer; QN_FANOUT:integer;
               VDD:real; TEMPERATURE:real);
      port (D:     in   bit;
            CLK:   in   bit;
            RESET: in   bit;
            Q:     out  bit;
            QBAR:  out  bit);

   end component;

   component hd_buff
      generic (DELAY:time;FANOUT:integer;VDD:real;
               TEMPERATURE:real;
               HD_BUFF_TABLE:cell_values:=work.tables.hd_buff_table);
      port (INPUT:  in   bit;
            OUTPUT: out  bit);

   end component;

   component inv
      generic (DELAY:time;FANOUT:integer;VDD:real;
               TEMPERATURE:real;
               INV_TABLE:cell_values:=work.tables.inv_table);
      port (INPUT:  in   bit;
            OUTPUT: out  bit);

   end component;

   signal VCC: bit:= '1';
   signal GND: bit:= '0';
```

```
signal I1301_Q, I1301_QN:      bit;
signal I1302_Q, I1302_QN:      bit;
signal I1303_Q, I1303_QN:      bit;
signal I1304_Q, I1304_QN:      bit;
signal I1305_Q, I1305_QN:      bit;
signal I1306_N, I1307_N:       bit;
signal I1308_Q, I1308_QN:      bit;
signal I1309_Q, I1309_QN:      bit;
signal I1310_Q, I1310_QN:      bit;
signal I1311_Q, I1311_QN:      bit;
signal I1312_Q, I1312_QN:      bit;
signal I1313_Q, I1313_QN:      bit;
signal I1314_Q, I1314_QN:      bit;
signal I1315_Q, I1315_QN:      bit;


for all: dff_pedg_rbar use entity work.dff_pedg_rbar(struct);
for all: hd_buff use entity work.hd_buff(behavioral);
for all: inv use entity work.inv(behavioral);


begin

I1301: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
          TEMPERATURE=> TEMPERATURE)
   port map (I1301_QN,clk13,I1307_N,open,I1301_QN);

I1302: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
          TEMPERATURE=> TEMPERATURE)
   port map (I1302_QN,I1301_QN,I1307_N,open,I1302_QN);

I1303: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
          TEMPERATURE=> TEMPERATURE)
   port map (I1303_QN,I1302_QN,I1307_N,open,I1303_QN);

I1304: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
          TEMPERATURE=> TEMPERATURE)
    port map (I1304_QN,I1303_QN,I1307_N,open,I1304_QN);

I1305: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
          TEMPERATURE=> TEMPERATURE)
    port map (I1305_QN,I1304_QN,I1307_N,open,I1305_QN);

I1306: inv
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
          TEMPERATURE=> TEMPERATURE)
    port map (res13, I1306_N);

I1307: hd_buff
```

```
        generic map (DELAY=>1 ns, FANOUT=> 13, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (I1306_N, I1307_N);

    I1308: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1308_QN,I1305_QN,I1307_N,open,I1308_QN);

    I1309: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1309_QN,I1308_QN,I1307_N,open,I1309_QN);

    I1310: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1310_QN,I1309_QN,I1307_N,open,I1310_QN);

    I1311: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1311_QN,I1310_QN,I1307_N,open,I1311_QN);

    I1312: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1312_QN,I1311_QN,I1307_N,open,I1312_QN);

    I1313: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1313_QN,I1312_QN,I1307_N,open,I1313_QN);

    I1314: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1314_QN,I1313_QN,I1307_N,open,I1314_QN);

    I1315: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1315_QN,I1314_QN,I1307_N,q13,I1315_QN);

end struct;
```

```
--use STD.Standard.all;
library gemlib;
use gemlib.tables.all;
use gemlib.gem_constants.all;
use gemlib.gem_delays.all;

entity rctr_18stage is

   generic (q18_fanout:       integer;
            testclock_fanout: integer;
            testoutput_fanout: integer;
            VDD:              real;
            TEMPERATURE:      real);

   port   (clk18, res18:                  in   bit;
           q18, testclock, testoutput:        out  bit);

end rctr_18stage;

architecture struct of rctr_18stage is

   component dff_pedg_rbar
      generic (Q_FANOUT: integer; QN_FANOUT:integer;
               VDD:real; TEMPERATURE:real);
      port (D:     in   bit;
            CLK:   in   bit;
            RESET: in   bit;
            Q:     out  bit;
            QBAR:  out  bit);

   end component;

   component hd_buff
      generic (DELAY:time;FANOUT:integer;VDD:real;
               TEMPERATURE:real;
               HD_BUFF_TABLE:cell_values:=gemlib.tables.hd_buff_table);
      port (INPUT:   in   bit;
            OUTPUT:  out  bit);

   end component;

   component inv
      generic (DELAY:time;FANOUT:integer;VDD:real;
               TEMPERATURE:real;
               INV_TABLE:cell_values:=gemlib.tables.inv_table);
      port (INPUT:   in   bit;
            OUTPUT:  out  bit);

   end component;

   signal VCC: bit:= '1';
   signal GND: bit:= '0';

   signal I1801_Q, I1801_QN:       bit;
```

```
    signal I1802_Q, I1802_QN:       bit;
    signal I1803_Q, I1803_QN:       bit;
    signal I1804_Q, I1804_QN:       bit;
    signal I1805_Q, I1805_QN:       bit;
    signal I1806_Q, I1806_QN:       bit;
    signal I1807_N, I1808_N:        bit;
    signal I1809_N:                 bit;
    signal I1810_Q, I1810_QN:       bit;
    signal I1811_Q, I1811_QN:       bit;
    signal I1812_Q, I1812_QN:       bit;
    signal I1813_Q, I1813_QN:       bit;
    signal I1814_Q, I1814_QN:       bit;
    signal I1815_Q, I1815_QN:       bit;
    signal I1816_Q, I1816_QN:       bit;
    signal I1817_Q, I1817_QN:       bit;
    signal I1818_Q, I1818_QN:       bit;
    signal I1819_Q, I1819_QN:       bit;
    signal I1820_Q, I1820_QN:       bit;
    signal I1821_Q, I1821_QN:       bit;

    for all: dff_pedg_rbar use entity work.dff_pedg_rbar(struct);
    for all: hd_buff use entity gemlib.hd_buff(behavioral);
    for all: inv use entity gemlib.inv(behavioral);


begin
    I1801: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1801_QN,I1807_N,I1808_N,open,I1801_QN);

    I1802: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1802_QN,I1801_QN,I1808_N,open,I1802_QN);

    I1803: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1803_QN,I1802_QN,I1808_N,open,I1803_QN);

    I1804: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1804_QN,I1803_QN,I1808_N,open,I1804_QN);

    I1805: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I1805_QN,I1804_QN,I1808_N,open,I1805_QN);

    I1806: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
                TEMPERATURE=> TEMPERATURE)
```

```
    port map (I1806_QN,I1805_QN,I1808_N,open,I1806_QN);

I1807: inv
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (clk18, I1807_N);

I1808: hd_buff
    generic map (DELAY=>1 ns, FANOUT=> 18, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (res18, I1808_N);

I1809: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1812_Q, I1809_N);

I1810: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1810_QN,I1806_QN,I1808_N,open,I1810_QN);

I1811: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1811_QN,I1810_QN,I1808_N,open,I1811_QN);

I1812: dff_pedg_rbar
    generic map (Q_FANOUT=> 2, QN_FANOUT=> 1, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1812_QN,I1811_QN,I1808_N,I1812_Q,I1812_QN);

I1813: dff_pedg_rbar
    generic map (Q_FANOUT=> testoutput_fanout, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1813_QN,I1809_N,I1808_N,testoutput,I1813_QN);

I1814: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1814_QN,I1813_QN,I1808_N,open,I1814_QN);

I1815: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1815_QN,I1814_QN,I1808_N,open,I1815_QN);

I1816: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1816_QN,I1815_QN,I1808_N,open,I1816_QN);

I1817: dff_pedg_rbar
    generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
```

```
            TEMPERATURE=> TEMPERATURE)
        port map (I1817_QN,I1816_QN,I1808_N,open,I1817_QN);

    I1818: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
        port map (I1818_QN,I1817_QN,I1808_N,open,I1818_QN);

    I1819: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
        port map (I1819_QN,I1818_QN,I1808_N,open,I1819_QN);

    I1820: dff_pedg_rbar
        generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
        port map (I1820_QN,I1819_QN,I1808_N,open,I1820_QN);

    I1821: dff_pedg_rbar
        generic map (Q_FANOUT=> q18_fanout, QN_FANOUT=> 1, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
        port map (I1821_QN,I1820_QN,I1808_N,q18,I1821_QN);

    testclock <= I1812_Q;

end struct;

use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-------------------------------------------------
-- Structural VHDL description of the CHP131 for
-- the PRC70 Radio.
-------------------------------------------------

entity CHP131A is

generic (TEMPERATURE:real;
 VDD:real);

port (PRCTR:inbit;
    DIR:inbit;
    EXDISN:inbit;
    CRSE:inbit;
    HOME:inbit;
    EXEC:inbit;
    SETDIR:inbit;
    RLEGFF:inbit;
    PULSEI:inbit;
    MODSAMPN:inbit;
    LCMD:inbit;
    ENPCTR:inbit;
    ENPSIG:inbit;
```

```
    P5U10:inbit;
    FINE:inbit;
    BNBUS:inbit_vector(3 downto 1);
    LLIMN:outbit;
    RLEG:outbit;
    RSTSW:outbit;
    PMRX:outbit;
    SENS:outbit;
    P8N:outbit;
    P4N:outbit;
    P2N:outbit;
    P1N:outbit;
    RRBUS:outbit_vector(7 downto 1);
    PRBUS:outbit_vector(3 downto 1);
    PHSEN:outbit);

end CHP131A;

architecture struct of chp131a is

  component cmos_in
      generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          CMOS_IN_TABLE:cell_values:=work.tables.cmos_in_table);
      port (INPUT:in bit;
          OUTPUT:outbit);
  end component;

  component cmos_pad
      generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          CMOS_PAD_TABLE:cell_values:=work.tables.cmos_pad_table);
      port (INPUT:in bit;
          OUTPUT:out bit);
  end component;

  component out_buff
      generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          OUT_BUFF_TABLE:cell_values:=work.tables.out_buff_table);
      port (INPUT:in bit;
          OUTPUT:out bit);
  end component;

  component cmos_5vout
      generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          CMOS_5VOUT_TABLE:cell_values:=work.tables.cmos_5vout_table);
      port (INPUT:in bit;
          EN:in bit;
          OUTPUT:out bit);
  end component;

  component inv
```

```
    generic (DELAY:time;FANOUT:integer;VDD:real;
        TEMPERATURE:real;
        INV_TABLE:cell_values:=work.tables.inv_table);
    port (INPUT:in bit;
        OUTPUT:out bit);
end component;

component and2
    generic (DELAY:time;FANOUT:integer;VDD:real;
        TEMPERATURE:real;
        AND2_TABLE:cell_values:=work.tables.and2_table);
    port (INPUT1:in bit;
        INPUT2:in bit;
        OUTPUT:out bit);
end component;

component and3
    generic (DELAY:time;FANOUT:integer;VDD:real;
        TEMPERATURE:real;
        AND3_TABLE:cell_values:=work.tables.and3_table);
    port (INPUT1:in bit;
        INPUT2:in bit;
        INPUT3:in bit;
        OUTPUT:out bit);
end component;

component nand2
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
     VDD:real;TEMPERATURE:real;
        NAND2_TABLE:cell_values:=work.tables.nand2_table);
    port (INPUT1:in bit;
        INPUT2:in bit;
        OUTPUT:out bit);
end component;

component nand3
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
     VDD:real;TEMPERATURE:real;
        NAND3_TABLE:cell_values:=work.tables.nand3_table);
    port (INPUT1:in bit;
        INPUT2:in bit;
        INPUT3:in bit;
        OUTPUT:out bit);
end component;

component nand4
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
     VDD:real;TEMPERATURE:real;
        NAND4_TABLE:cell_values:=work.tables.nand4_table);
    port (INPUT1:in bit;
        INPUT2:in bit;
        INPUT3:in bit;
        INPUT4:in bit;
        OUTPUT:out bit);
```

```
end component;

component nor2
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
     VDD:real;TEMPERATURE:real;
         NOR2_TABLE:cell_values:=work.tables.nor2_table);
    port (INPUT1:in  bit;
        INPUT2:in  bit;
        OUTPUT:out bit);
end component;

component nor3
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
     VDD:real;TEMPERATURE:real;
         NOR3_TABLE:cell_values:=work.tables.nor3_table);
    port (INPUT1:in  bit;
        INPUT2:in  bit;
        INPUT3:in  bit;
        OUTPUT:out bit);
end component;

component nor4
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:= 0 fs;
     VDD:real;TEMPERATURE:real;
         NOR4_TABLE:cell_values:=work.tables.nor4_table);
    port (INPUT1:in  bit;
        INPUT2:in  bit;
        INPUT3:in  bit;
        INPUT4:in  bit;
        OUTPUT:out bit);
end component;

component exor
    generic (DELAY:time;FANOUT:integer;VDD:real;
         TEMPERATURE:real;
         EXOR_TABLE:cell_values:=work.tables.exor_table);
    port (INPUT1:in  bit;
        INPUT2:in  bit;
        OUTPUT:out bit);
end component;

component dff_pedg_rbar
    generic (Q_FANOUT:integer;QN_FANOUT:integer;
     VDD:real;TEMPERATURE:real);
    port (D:inbit;
        CLK:inbit;
        RESET:inbit;
        Q:outbit;
        QBAR:outbit);
end component;

component up5ctr
    generic (A_FANOUT:integer;B_FANOUT:integer;
         C_FANOUT:integer;D_FANOUT:integer;E_FANOUT:integer;
```

```
      VDD:real;TEMPERATURE:real);
    port (CLK:inbit;
        RST:inbit;
        A:outbit;
        B:outbit;
        C:outbit;
        D:outbit;
        E:outbit);
end component;

component syn4ctr
    generic (A_FANOUT:integer;B_FANOUT:integer;
        C_FANOUT:integer;D_FANOUT:integer;CO_FANOUT:integer;
     VDD:real;TEMPERATURE:real);
    port (CLK:inbit;
        UP:inbit;
        RST:inbit;
        CI:inbit;
        A:inoutbit;
        B:outbit;
        C:outbit;
        D:outbit;
        CO:outbit);
end component;

component aor
    generic (A0_FANOUT:integer;B0_FANOUT:integer;
     C0_FANOUT:integer;D0_FANOUT:integer;
     VDD:real;TEMPERATURE:real);
    port (A1, A2:in bit;
        B1, B2:in bit;
        C1, C2:in bit;
        D1, D2:in bit;
        K1, K2:in bit;
        A0, B0, C0, D0:out bit);
end component;

signal VCC: bit:= '1';
signal GND: bit:= '0';

signal I909_O, I910_O, I911_O, I912_O, I913_O:bit;
signal I915_O, I916_O, I917_O, I918_O:bit;
signal I919_O, I920_O, I921_O, I922_O, I923_O:bit;
signal I151_O, I152_Q, I152_QN, I153_O, I154_O:bit;
signal I155_O, I156_O, I157_O, I158_O, I159_O:bit;
signal I160_A, I160_B, I160_C, I160_D, I160_CO:bit;
signal I161_A, I161_B, I161_C, I161_D, I161_CO:bit;
signal I162_A, I162_B, I162_C, I162_D, I162_CO:bit;
signal I163_Q, I163_QN, I164_Q, I164_QN, I167_O:bit;
signal I168_O, I169_O, I170_O, I171_O, I172_O:bit;
signal I173_O, I174_O, I178_O, I179_O, I180_O:bit;
signal I181_O, I182_O, I183_O, I184_O, I192_O:bit;
signal I193_O, I195_O, I197_O, I198_O, I199_O:bit;
signal I200_O, I204_O, I208_O, I209_O, I210_O:bit;
```

```
signal I211_O, I212_O, I213_O, I214_O, I215_O:bit;
signal I216_O, I217_O, I219_O, I220_O, I221_O:bit;
signal I222_O, I223_O, I224_O, I225_O, I226_O:bit;
signal I227_O, I228_O, I229_O, I232_O, I965_O:bit;
signal I962_O, I963_O, I964_O, I196_O:bit;
signal I966_O, I967_O, I70_Q, I70_QN, I71_Q:bit;
signal I71_QN, I72_O, I78_Q, I78_QN, I94_O:bit;
signal I41_O, I42_O, I43_O, I44_O, I45_O, I46_O:bit;
signal I47_O, I48_O, I56_O, I58_O, I62_O, I63_O:bit;
signal I64_O, I24_O, I25_O, I26_O, I27_O, I401_O:bit;
signal I4_A, I4_B, I4_C, I4_D, I4_E:bit;
signal I5_A, I5_B, I5_C, I5_D, I5_E:bit;
signal I6_A0, I6_B0, I6_C0, I6_D0:bit;
signal I7_A, I7_B, I7_C, I7_D, I7_E:bit;
signal I61_Q, I61_QN, I77_O, I54_O:bit;
signal I2068_O, I2069_O, I2070_O, I2071_O:bit;
signal I2072_O, I2073_O, I2074_O, I2075_O:bit;
signal I2076_O, I2077_O, I2067_O, I2061_O:bit;
signal I2062_O, I2064_O, I2065_O, I2066_O:bit;


for all:cmos_in use entity work.cmos_in(behavioral);
for all:cmos_pad use entity work.cmos_pad(behavioral);
for all:out_buff use entity work.out_buff(behavioral);
for all:cmos_5vout use entity work.cmos_5vout(behavioral);
for all:inv use entity work.inv(behavioral);
for all:and2 use entity work.and2(behavioral);
for all:and3 use entity work.and3(behavioral);
for all:nand2 use entity work.nand2(behavioral);
for all:nand3 use entity work.nand3(behavioral);
for all:nand4 use entity work.nand4(behavioral);
for all:nor2 use entity work.nor2(behavioral);
for all:nor3 use entity work.nor3(behavioral);
for all:nor4 use entity work.nor4(behavioral);
for all:exor use entity work.exor(behavioral);
for all:dff_pedg_rbar use entity work.dff_pedg_rbar(struct);
for all:up5ctr use entity work.up5ctr(struct);
for all:syn4ctr use entity work.syn4ctr(struct);
for all:aor use entity work.aor(struct);


begin


I923: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>3, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (PRCTR, I923_O);

I922: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>8, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (DIR, I922_O);
```

```
I921: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (EXDISN, I921_O);

I917: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (CRSE, I917_O);

I918: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (HOME, I918_O);

I919: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (EXEC, I919_O);

I920: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (SETDIR, I920_O);

I916: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (RLEGFF, I916_O);

I909: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>3, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (PULSEI, I909_O);

I910: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (MODSAMPN, I910_O);

I911: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (LCMD, I911_O);

I912: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (ENPCTR, I912_O);

I913: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (ENPSIG, I913_O);
```

```
I915: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (FINE, I915_O);

I151: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I919_O, I151_O);

I152: dff_pedg_rbar
        generic map (Q_FANOUT=>1, QN_FANOUT=>1, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
        port map (I152_QN, I919_O, I918_O, I152_Q, I152_QN);

I153: nor3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I921_O, I209_O, I156_O, I153_O);

I154: nor3
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I157_O, I209_O, I921_O, I154_O);

I155: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I918_O, I155_O);

I156: exor
        generic map (DELAY=> 1 ns, FANOUT=>13, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I152_Q, I916_O, I156_O);

I157: inv
        generic map (DELAY=> 1 ns, FANOUT=>9, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I156_O, I157_O);

I158: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I920_O, I157_O, I158_O);

I159: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I156_O, I920_O, I159_O);

I160: syn4ctr
        generic map (A_FANOUT=>5, B_FANOUT=>2, C_FANOUT=>5,
          D_FANOUT=>1, CO_FANOUT=>1, VDD=>VDD,
```

```
        TEMPERATURE=>TEMPERATURE)
        port map (I153_O, I922_O, I923_O, VCC,
        I160_A, I160_B, I160_C, I160_D, I160_CO);

I161: syn4ctr
        generic map (A_FANOUT=>6, B_FANOUT=>5, C_FANOUT=>3,
          D_FANOUT=>2, CO_FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
        port map (I154_O, I922_O, I923_O, VCC,
        I161_A, I161_B, I161_C, I161_D, I161_CO);

I162: syn4ctr
        generic map (A_FANOUT=>3, B_FANOUT=>3, C_FANOUT=>3,
          D_FANOUT=>1, CO_FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
        port map (I154_O, I922_O, I923_O, I161_CO,
        I162_A, I162_B, I162_C, I162_D, I162_CO);

I163: dff_pedg_rbar
        generic map (Q_FANOUT=>1, QN_FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
        port map (I922_O, I158_O, VCC, I163_Q, I163_QN);

I164: dff_pedg_rbar
        generic map (Q_FANOUT=>1, QN_FANOUT=>1, VDD=>VDD,
        TEMPERATURE=>TEMPERATURE)
        port map (I922_O, I159_O, VCC, I164_Q, I164_QN);

I167: nor3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I160_A, I160_B, I160_C, I167_O);

I168: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_B, I161_A, I168_O);

I169: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_D, I161_C, I169_O);

I170: exor
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I922_O, I163_Q, I170_O);

I171: exor
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I922_O, I164_Q, I171_O);
```

```
I172: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I167_O, I172_O);

I173: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I169_O, I168_O, I173_O);

I174: nand3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I162_C, I162_B, I162_A, I174_O);

I178: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I173_O, I178_O);

I182: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I174_O, I178_O, I182_O);

I2068: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I162_A, I2068_O);

I2141: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2068_O, RRBUS(5));

I2069: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I162_B, I2069_O);

I900: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2069_O, RRBUS(6));

I2070: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I162_C, I2070_O);

I899: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2070_O, RRBUS(7));
```

```
I2071: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_A, I2071_O);

I902: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2071_O, RRBUS(1));

I2072: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_B, I2072_O);

I903: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2072_O, RRBUS(2));

I2073: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_C, I2073_O);

I904: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2073_O, RRBUS(3));

I2074: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_D, I2074_O);

I905: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2074_O, RRBUS(4));

I2075: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I160_A, I2075_O);

I906: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2075_O, PRBUS(1));

I2076: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I160_B, I2076_O);
```

```
I907: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2076_O, PRBUS(2));

I2077: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I160_C, I2077_O);

I908: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2077_O, PRBUS(3));

I192: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I160_A, I192_O);

I195: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I192_O, I160_C, I195_O);

I200: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I195_O, I160_C, I200_O);

I208: and3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I200_O, I160_B, I157_O, I208_O);

I2067: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I208_O, I2067_O);

I895: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2067_O, PHSEN);

I198: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (GND, I160_A, I198_O);

I197: nand3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I160_A, I160_B, I160_C, I197_O);
```

```
I204: and2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I198_O, I197_O, I204_O);

I962: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (BNBUS(2), I962_O);

I963: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (BNBUS(3), I963_O);

I964: cmos_in
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (BNBUS(1), I964_O);

I965: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I964_O, I965_O);

I966: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I963_O, I966_O);

I967: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I962_O, I967_O);

I210: nand3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_A, I161_B, I967_O, I210_O);

I211: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_A, I966_O, I211_O);

I212: nand4
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I965_O, I161_B, I161_C, I161_A, I212_O);

I213: nand3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I212_O, I211_O, I210_O, I213_O);
```

```
I214: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I182_O, I213_O, I214_O);

I215: nor3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I161_C, I161_B, I161_A, I215_O);

I216: nor4
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I162_C, I162_B, I162_A, I161_D, I216_O);

I217: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I216_O, I215_O, I217_O);

I219: inv
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I922_O, I219_O);

I220: nor3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I219_O, I214_O, I48_O, I220_O);

I221: nor3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I219_O, I48_O, I204_O, I221_O);

I222: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I220_O, I156_O, I222_O);

I223: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I221_O, I157_O, I223_O);

I224: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I217_O, I157_O, I224_O);

I225: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I156_O, I172_O, I225_O);
```

```
I226: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I222_O, I223_O, I226_O);

I227: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I224_O, I225_O, I227_O);

I228: nor3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I922_O, I48_O, I227_O, I228_O);

I229: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I228_O, I226_O, I229_O);

I232: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I917_O, I232_O);

I2062: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I156_O, I2062_O);

I893: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2062_O, RLEG);

I2061: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I229_O, I2061_O);

I894: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2061_O, LLIMN);

I179: nand2
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I170_O, I157_O, I179_O);

I180: nand2
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I156_O, I171_O, I180_O);
```

```
I181: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I45_O, I181_O);

I183: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I184_O, I71_Q, I183_O);

I184: nor3
        generic map (DELAY=> 1 ns, FANOUT=>3, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I917_O, I181_O, I183_O, I184_O);

I193: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I180_O, I179_O, I193_O);

I196: nor2
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I199_O, I193_O, I196_O);

I199: nand2
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I155_O, I184_O, I199_O);

I209: nand2
        generic map (DELAY=> 1 ns, FANOUT=>3, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I199_O, I911_O, I209_O);

I94: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I909_O, I94_O);

I78: dff_pedg_rbar
        generic map (Q_FANOUT=>2, QN_FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (VCC, I94_O, I910_O, I78_Q, I78_QN);

I41: nor3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I156_O, I196_O, I78_Q, I41_O);

I44: nand3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I157_O, I179_O, I912_O, I44_O);
```

```
I42: nor3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I157_O, I196_O, I78_Q, I42_O);

I43: nand3
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I156_O, I912_O, I180_O, I43_O);

I4: up5ctr
        generic map (A_FANOUT=>1, B_FANOUT=>1, C_FANOUT=>1,
           D_FANOUT=>1, E_FANOUT=>1, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
        port map (I44_O, I41_O, I4_A, I4_B, I4_C, I4_D, I4_E);

I5: up5ctr
        generic map (A_FANOUT=>1, B_FANOUT=>1, C_FANOUT=>1,
           D_FANOUT=>1, E_FANOUT=>1, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
        port map (I43_O, I42_O, I5_A, I5_B, I5_C, I5_D, I5_E);

I45: nor2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I4_E, I5_E, I45_O);

I6: aor
        generic map (A0_FANOUT=>3, B0_FANOUT=>3,
             C0_FANOUT=>3, D0_FANOUT=>3,
            VDD=>VDD, TEMPERATURE=>TEMPERATURE)
        port map (I4_A, I5_A, I4_B, I5_B, I4_C, I5_C, I4_D, I5_D,
        I157_O, I156_O, I6_A0, I6_B0, I6_C0, I6_D0);

I72: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I184_O, I71_Q, I72_O);

I70: dff_pedg_rbar
        generic map (Q_FANOUT=>2, QN_FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (VCC, I209_O, I72_O, I70_Q, I70_QN);

I71: dff_pedg_rbar
        generic map (Q_FANOUT=>2, QN_FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I70_Q, I151_O, I70_Q, I71_Q, I71_QN);

I54: nand4
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I6_A0, I6_B0, I6_C0, I6_D0, I54_O);
```

```
I46: nor4
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I6_A0, I6_B0, I6_C0, I6_D0, I46_O);

I63: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I913_O, I63_O);

I64: nand2
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I232_O, I54_O, I64_O);

I62: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I63_O, I156_O, I62_O);

I48: inv
        generic map (DELAY=> 1 ns, FANOUT=>3, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I64_O, I48_O);

I56: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I46_O, I56_O);

I77: nand4
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I909_O, I56_O, I156_O, I61_QN, I77_O);

I47: nand4
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I61_QN, I156_O, I909_O, I46_O, I47_O);

I58: nand3
        generic map (DELAY=> 1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I62_O, I77_O, I915_O, I58_O);

I61: dff_pedg_rbar
        generic map (Q_FANOUT=>1, QN_FANOUT=>2, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
        port map (I7_C, I911_O, I401_O, I61_Q, I61_QN);

I401: inv
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I58_O, I401_O);
```

```
I7: up5ctr
        generic map (A_FANOUT=>1, B_FANOUT=>1, C_FANOUT=>1,
            D_FANOUT=>1, E_FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I47_O, I58_O, I7_A, I7_B, I7_C, I7_D, I7_E);

I24: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I913_O, I6_D0, I24_O);

I25: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I913_O, I6_C0, I25_O);

I26: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I913_O, I6_B0, I26_O);

I27: nand2
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I913_O, I6_A0, I27_O);

I2064: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I70_QN, I2064_O);

I890: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2064_O, RSTSW);

I2065: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I64_O, I2065_O);

I891: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2065_O, PMRX);

I2066: out_buff
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I61_Q, I2066_O);
I892: cmos_pad
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I2066_O, SENS);
```

```
  I884: cmos_5vout
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I24_O, P5U10, P8N);

  I885: cmos_5vout
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I25_O, P5U10, P4N);

  I886: cmos_5vout
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I26_O, P5U10, P2N);

  I887: cmos_5vout
        generic map (DELAY=> 1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
        port map (I27_O, P5U10, P1N);

end struct;


use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-------------------------------------------------------------
-- File which contains the operating voltage and temperature
-- parameters which the CHP131 chip is simulated at. These
-- parameters are passed to the CHP131.vhd file during simulation
-- via the generic statements.
-------------------------------------------------------------

entity chp131amod is

port (PRCTR:inbit;
    DIR:inbit;
    EXDISN:inbit;
    CRSE:inbit;
    HOME:inbit;
    EXEC:inbit;
    SETDIR:inbit;
    RLEGFF:inbit;
    PULSEI:inbit;
    MODSAMPN:inbit;
    LCMD:inbit;
    ENPCTR:inbit;
    ENPSIG:inbit;
    P5U10:inbit;
    FINE:inbit;
    BNBUS:inbit_vector(3 downto 1);
    LLIMN:outbit;
    RLEG:outbit;
```

```
    RSTSW:outbit;
    PMRX:outbit;
    SENS:outbit;
    P8N:outbit;
    P4N:outbit;
    P2N:outbit;
    P1N:outbit;
    RRBUS:outbit_vector(7 downto 1);
    PRBUS:outbit_vector(3 downto 1);
    PHSEN:outbit);

end chp131amod;


architecture chp131a of chp131amod is

  component chp131a

generic (TEMPERATURE:real; VDD:real);

port (PRCTR:inbit;
    DIR:inbit;
    EXDISN:inbit;
    CRSE:inbit;
    HOME:inbit;
    EXEC:inbit;
    SETDIR:inbit;
    RLEGFF:inbit;
    PULSEI:inbit;
    MODSAMPN:inbit;
    LCMD:inbit;
    ENPCTR:inbit;
    ENPSIG:inbit;
    P5U10:inbit;
    FINE:inbit;
    BNBUS:inbit_vector(3 downto 1);
    LLIMN:outbit;
    RLEG:outbit;
    RSTSW:outbit;
    PMRX:outbit;
    SENS:outbit;
    P8N:outbit;
    P4N:outbit;
    P2N:outbit;
    P1N:outbit;
    RRBUS:outbit_vector(7 downto 1);
    PRBUS:outbit_vector(3 downto 1);
    PHSEN:outbit);
end component;

for all: chp131a use entity work.chp131a(struct);

begin
```

```
N0:chp131a

generic map (TEMPERATURE=> 298.15, VDD=> 10.0)

port map (PRCTR, DIR, EXDISN, CRSE, HOME, EXEC,
 SETDIR, RLEGFF, PULSEI, MODSAMPN, LCMD,
 ENPCTR, ENPSIG, P5U10, FINE, BNBUS, LLIMN,
 RLEG, RSTSW, PMRX, SENS, P8N, P4N, P2N,
 P1N, RRBUS, PRBUS, PHSEN) ;

end chp131a;


use work.tables.all;
use work.gen_constants.all;
use work.gen_delays.all;

-----------------------------------------------
-- Test Bench for the simulation of the CHP131 chip in
-- the PRC70 radio.
-----------------------------------------------

entity test_chp131a is
end test_chp131a;


architecture tchp131a of test_chp131a is

-- Input Signals

signal PRCTR:bit:= '0';
signal DIR:bit:= '1';
signal EXDISN:bit:= '1';
signal CRSE:bit:= '1';
signal HOME:bit:= '0';
signal EXEC:bit:= '0';
signal SETDIR:bit:= '0';
signal RLEGFF:bit:= '1';
signal PULSEI:bit:= '0';
signal MODSAMPN:bit:= '0';
signal LCMD:bit:= '1';
signal ENPCTR:bit:= '0';
signal ENPSIG:bit:= '0';
signal P5U10:bit:= '1';
signal FINE:bit:= '0';
signal BNBUS:bit_vector(3 downto 1):= "111";

-- Output Signals

signal LLIMN:bit;
signal RLEG:bit;
signal RSTSW:bit;
signal PMRX:bit;
signal SENS:bit;
```

```
signal P8N:bit;
signal P4N:bit;
signal P2N:bit;
signal P1N:bit;
signal RRBUS:bit_vector(7 downto 1);
signal PRBUS:bit_vector(3 downto 1);
signal PHSEN:bit;

-- Simulation Signals which simulator uses to enable the
-- clocks and to terminate the simulation.

signal L1:bit;
signal L2:bit;
signal EXDISNA:bit:= '1';
signal ENPCTRA:bit:= '0';
signal STOP:bit:= '0';


component chp131amod
port (PRCTR:inbit;
    DIR:inbit;
    EXDISN:inbit;
    CRSE:inbit;
    HOME:inbit;
    EXEC:inbit;
    SETDIR:inbit;
    RLEGFF:inbit;
    PULSEI:inbit;
    MODSAMPN:inbit;
    LCMD:inbit;
    ENPCTR:inbit;
    ENPSIG:inbit;
    P5U10:inbit;
    FINE:inbit;
    BNBUS:inbit_vector(3 downto 1);
    LLIMN:outbit;
    RLEG:outbit;
    RSTSW:outbit;
    PMRX:outbit;
    SENS:outbit;
    P8N:outbit;
    P4N:outbit;
    P2N:outbit;
    P1N:outbit;
    RRBUS:outbit_vector(7 downto 1);
    PRBUS:outbit_vector(3 downto 1);
    PHSEN:outbit);
end component;

for DC: chp131amod use entity work.chp131amod(chp131a);

begin

BNBUS <= "111" after 0 ns,
```

```
        "011" after 45800 ns, "111" after 46000 ns,
            "101" after 47000 ns, "111" after 47200 ns,
            "110" after 49000 ns, "111" after 49200 ns;

PRCTR <= '1' after 200 ns, '0' after 400 ns,
        '1' after 103400 ns, '0' after 103600 ns;

DIR <= '0' after 52600 ns, '1' after 103800 ns,
       '0' after 106800 ns, '1' after 124800 ns,
       '0' after 125400 ns, '1' after 125600 ns;

CRSE <= '0' after 111400 ns, '1' after 136400 ns;

HOME <= '1' after 111200 ns, '0' after 112400 ns,
        '1' after 124400 ns, '0' after 126000 ns,
        '1' after 127200 ns;

EXEC <= '1' after 111800 ns, '0' after 112000 ns,
        '1' after 112600 ns, '0' after 112800 ns,
        '1' after 126800 ns, '0' after 127000 ns,
        '1' after 136000 ns, '0' after 136200 ns;

SETDIR <= '1' after 110200 ns, '0' after 110400 ns,
          '1' after 110800 ns, '0' after 111000 ns,
          '1' after 113800 ns, '0' after 114000 ns,
          '1' after 125000 ns, '0' after 125200 ns,
          '1' after 128400 ns, '0' after 128600 ns;

RLEGFF <= '0' after 103200 ns, '1' after 110000 ns,
          '0' after 110600 ns, '1' after 113600 ns,
          '0' after 127400 ns;

PULSEI <= '1' after 113200 ns, '0' after 113400 ns,
          '1' after 114600 ns, '0' after 114800 ns,
          '1' after 115000 ns, '0' after 115200 ns,
          '1' after 115400 ns, '0' after 115600 ns,
          '1' after 115800 ns, '0' after 116800 ns,
          '1' after 117000 ns, '0' after 117200 ns,
          '1' after 117400 ns, '0' after 117600 ns,
          '1' after 117800 ns, '0' after 118000 ns,
          '1' after 118200 ns, '0' after 128000 ns,
          '1' after 128200 ns;

MODSAMPN <= '1' after 113000 ns, '0' after 126400 ns,
            '1' after 126600 ns, '0' after 127600 ns,
            '1' after 127800 ns, '0' after 135600 ns,
            '1' after 135800 ns;

LCMD <= '0' after 111600 ns, '1' after 112200 ns,
        '0' after 116000 ns, '1' after 116200 ns,
        '0' after 118400 ns, '1' after 118600 ns,
        '0' after 126200 ns, '1' after 135200 ns,
        '0' after 135400 ns;
```

```
ENPSIG <= '1' after 114200 ns, '0' after 118800 ns,
      '1' after 124600 ns;

FINE <= '1' after 114400 ns, '0' after 116400 ns,
     '1' after 116600 ns;

L1 <= '1' after 600 ns, '0' after 45600 ns,
    '1' after 47400 ns, '0' after 48800 ns,
    '1' after 49400 ns, '0' after 52400 ns,
    '1' after 52800 ns, '0' after 103000 ns,
    '1' after 104000 ns, '0' after 106600 ns,
    '1' after 107000 ns, '0' after 108800 ns;

L2 <= '1' after 118800 ns, '0' after 124200 ns,
    '1' after 128600 ns, '0' after 134700 ns;

EXDISNA <= '0' after 600 ns,
   '1' after 45600 ns, '0' after 46200 ns,
   '1' after 46400 ns, '0' after 46600 ns,
   '1' after 46800 ns, '0' after 47400 ns,
   '1' after 47600 ns, '0' after 47800 ns,
   '1' after 48000 ns, '0' after 48600 ns,
   '1' after 48800 ns, '0' after 49400 ns,
   '1' after 52400 ns, '0' after 52800 ns,
   '1' after 103000 ns, '0' after 104000 ns,
   '1' after 106600 ns, '0' after 107000 ns,
   '1' after 108800 ns, '0' after 109400 ns,
   '1' after 109600 ns, '0' after 109800 ns,
   '1' after 118600 ns;

ENPCTRA <= '1' after 124200 ns, '0' after 125800 ns,
      '0' after 135000 ns, '0' after 135200 ns;

STOP <= '1' after 160000 ns;

DC: chp131amod port map (PRCTR, DIR, EXDISN, CRSE, HOME, EXEC,
 SETDIR, RLEGFF, PULSEI, MODSAMPN, LCMD,
 ENPCTR, ENPSIG, P5U10, FINE, BNBUS, LLIMN,
 RLEG, RSTSW, PMRX, SENS, P8N, P4N, P2N,
 P1N, RRBUS, PRBUS, PHSEN) ;


-- This process genarates the EXDISN clock. The clock is
-- gated with the L1 signal to disable clocking where
-- necessary.

P1:process
begin

EXDISN <= EXDISNA ;
wait for 1 fs;

while L1 = '1' loop
 EXDISN <= '0';
```

```
 wait for 200 ns;
 EXDISN <= '1';
 wait for 200 ns;
end loop;

wait on L1, EXDISNA;

end process P1;

-- This process genarates the ENPCTR clock. The clock is
-- gated with the L1 signal to disable clocking where
-- necessary.

P2:process
begin

wait for 1 fs;
ENPCTR <= ENPCTRA ;

while L2 = '1' loop
 ENPCTR <= '0';
 wait for 200 ns;
 ENPCTR <= '1';
 wait for 200 ns;
end loop;

wait on L2, ENPCTRA;

end process P2;

P3:process
begin

    wait until STOP = '1';
    assert stop = '0' report "End of Simulation"
    severity FAILURE;

end process P3;

end tchp131a ;


use STD.Standard.all;
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

entity chp395 is

    generic (VDD:         real;
          TEMPERATURE:  real);

    port   (VSWR, TUNECMD, FREQCH, HOLD, HOME:        in    bit;
          SETDIR, CA, PULSEOFF, RSTPUL:             in    bit;
```

```
        TESTCLOCK, CB, CC, TESTOUTPUT, TUNERST:    out  bit;
        FCHFFN, TC1, TUNECYC, NOTUNE, CLOCK:       out  bit;
        EXHOLDN, PULSE5, PULSE1:                   out  bit);

end chip395;

architecture struct of chip395 is

-------------------------------------------------------------------
-- All components obtained from GEM gate behavioral descriptions written by --
-- Ken Keyes.  Each component gets its delay values from a table and packages--
-- which calculate delay values based on fanout, VDD, and temperature passed --
-- in through generic maps.                                      --
-------------------------------------------------------------------

  component cmos_in
      generic (DELAY:time;FANOUT:integer;VDD:real;
            TEMPERATURE:real;
            CMOS_IN_TABLE:cell_values:=work.tables.cmos_in_table);
        port (INPUT:  in   bit;
              OUTPUT: out  bit);
  end component;

  component cmos_pad
       generic (DELAY:time;FANOUT:integer;VDD:real;
            TEMPERATURE:real;
            CMOS_PAD_TABLE:cell_values:=work.tables.cmos_pad_table);
       port (INPUT:  in   bit;
             OUTPUT: out  bit);
  end component;

  component dff_pedg_rbar
       generic (Q_FANOUT: integer; QN_FANOUT:integer;
            VDD:real; TEMPERATURE:real);
       port (D:     in   bit;
             CLK:   in   bit;
             RESET: in   bit;
             Q:     out  bit;
             QBAR:  out  bit);
  end component;

  component dff_pedg_srbar
       generic (Q_FANOUT: integer; QN_FANOUT:integer;
            VDD:real; TEMPERATURE:real);
       port (D:     in   bit;
             CLK:   in   bit;
             RESET: in   bit;
             SET:   in   bit;
             Q:     out  bit;
             QBAR:  out  bit);
  end component;
```

```
component hd_buff
    generic (DELAY:time;FANOUT:integer;VDD:real;
        TEMPERATURE:real;
        HD_BUFF_TABLE:cell_values:=work.tables.hd_buff_table);
    port (INPUT:  in  bit;
       OUTPUT:  out  bit);
end component;

component inv
    generic (DELAY:time;FANOUT:integer;VDD:real;
        TEMPERATURE:real;
        INV_TABLE:cell_values:=work.tables.inv_table);
    port (INPUT:  in  bit;
       OUTPUT:  out  bit);
end component;

component nand2
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0fs;
       VDD:real;TEMPERATURE:real;
       NAND2_TABLE:cell_values:=work.tables.nand2_table);
    port (INPUT1:  in  bit;
       INPUT2:  in  bit;
       OUTPUT:  out  bit);
end component;

component nor2
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0fs;
          VDD:real;TEMPERATURE:real;
       NOR2_TABLE:cell_values:=work.tables.nor2_table);
    port (INPUT1:  in  bit;
       INPUT2:  in  bit;
       OUTPUT:  out  bit);
end component;

component nor3
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0fs;
          VDD:real;TEMPERATURE:real;
       NOR3_TABLE:cell_values:=work.tables.nor3_table);
    port (INPUT1:  in  bit;
       INPUT2:  in  bit;
       INPUT3:  in  bit;
       OUTPUT:  out  bit);
end component;

component out_buff
    generic (DELAY:time;FANOUT:integer;VDD:real;
       TEMPERATURE:real;
       OUT_BUFF_TABLE:cell_values:=work.tables.out_buff_table);
    port (INPUT:  in  bit;
       OUTPUT:  out  bit);
end component;
```

```
component rctr_9stage
    generic (q9_fanout:  integer;
             VDD:        real;
             TEMPERATURE: real);
    port (clk9, res9:    in     bit;
          q9:            out    bit);
end component;

component rctr_13stage
    generic (q13_fanout: integer;
             VDD:        real;
             TEMPERATURE: real);
    port (clk13, res13:  in     bit;
          q13:           out    bit);
end component;

component rctr_18stage
    generic (q18_fanout:       integer;
             testclock_fanout: integer;
             testoutput_fanout: integer;
             VDD:              real;
             TEMPERATURE:      real);
    port (clk18, res18:                in     bit;
          q18, testclock, testoutput:  out    bit);
end component;

-- ------------------------------------------------------------
-- All signal names correspond to the MIF file part names (except for the  --
-- $ sign which is a reserved character in VHDL) for easy cross reference  --
-- with the design drawings which were produced from a mentor graphics     --
-- workstation. Each part has an _N afterward to indicate the output signal --
-- of each part in their respective port map                                --
-- ------------------------------------------------------------

  signal VCC: bit:= '1';
  signal GND: bit:= '0';

  signal I1081_N,I1082_N,I1111_Q,I1111_QN :      bit;
  signal I270_N,I179_N,I2222_Q,I2222_QN   :      bit;
  signal I269_N,I382_N,I195_N,I184_N,I183_N :    bit;
  signal I189_N,I298_N,RCTR_18TC,RCTR_18Q18 :    bit;
  signal RCTR_18TO,I296_N,I295_N,I686_N   :      bit;
  signal I294_N,I1682_N,I1683_N,I1684_N   :      bit;
  signal I192_N,I193_N,I1686_N,I1041_N    :      bit;
  signal I1084_N,I1085_N,I1086_N,I1061_N  :      bit;
  signal I321_N,I168_N,I169_N,I172_N      :      bit;
  signal I3333_Q,I3333_QN,I173_N,I204_N   :      bit;
  signal I205_N,I227_N,I167_N,I166_N,I300_N :    bit;
  signal I1685_N,I1087_N,I1693_N,I188_N   :      bit;
  signal I1088_N,I1089_N,I1692_N,I299_N   :      bit;
  signal I151_N, I152_N,I155_N,I147_N     :      bit;
  signal I1090_N,I4444_Q,I4444_QN,I5555_Q :      bit;
  signal I5555_QN,I187_N,I292_N,I1095_N   :      bit;
  signal I1696_N,I290_N,I291_N,I1698_N    :      bit;
```

```
    signal I293_N,I1687_N,I1688_N,I287_N    :      bit;
    signal RCTR_13Q13,RCTR_9Q9,I1091_N,I272_N :      bit;
    signal I138_N,I281_N,I6666_Q,I6666_QN   :      bit;
    signal I136_N,I139_N,I140_N,I283_N      :      bit;
    signal I284_N,I1689_N,I285_N,I301_N     :      bit;
    signal I1690_N,I1099_N,I1100_N          :      bit;
    signal I273_N,I289_N,I1691_N,I1590_N    :      bit;

    for all: cmos_in use entity work.cmos_in(behavioral);
    for all: cmos_pad use entity work.cmos_pad(behavioral);
    for all: dff_pedg_rbar use entity work.dff_pedg_rbar(struct);
    for all: dff_pedg_srbar use entity work.dff_pedg_srbar(struct);
    for all: hd_buff use entity work.hd_buff(behavioral);
    for all: inv use entity work.inv(behavioral);
    for all: nand2 use entity work.nand2(behavioral);
    for all: nor2 use entity work.nor2(behavioral);
    for all: nor3 use entity work.nor3(behavioral);
    for all: out_buff use entity work.out_buff(behavioral);
    for all: rctr_9stage use entity work.rctr_9stage(struct);
    for all: rctr_13stage use entity work.rctr_13stage(struct);
    for all: rctr_18stage use entity work.rctr_18stage(struct);

begin

    I1081: cmos_in
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (VSWR, I1081_N);

    I1082: cmos_in
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (TUNECMD, I1082_N);

    I1111: dff_pedg_rbar
       generic map (Q_FANOUT=> 2, QN_FANOUT=> 1, VDD=> VDD,
              TEMPERATURE=> TEMPERATURE)
       port map (I1081_N, I1082_N, I270_N, I1111_Q, I1111_QN);

    I270: inv
        generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
                TEMPERATURE=> TEMPERATURE)
        port map (I189_N, I270_N);

    I179: nor2
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (I1111_Q,I1085_N,I179_N);

    I269: inv
        generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
                TEMPERATURE=>TEMPERATURE)
        port map (I179_N, I269_N);
```

```
I2222: dff_pedg_arbar
    generic map (Q_FANOUT=> 2, QN_FANOUT=> 1, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (GND, I269_N, I382_N, I1111_QN,I2222_Q, I2222_QN);

I382: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I3333_Q,I382_N);

I195: inv
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I2222_Q,I195_N);

I184: nor2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I2222_QN,I166_N,I184_N);

-------------------------------------------------------------------
-- Some gate instantiations required an initial delay because they were paired-
-- with other gates to form a Flip-Flop pattern that would have an undefined --
-- state when simulation began.  To offset the race condition in the feedback --
-- loop one of the gates in the pattern was given a 1 fs delay to wait until  --
-- the other had received and responded to its input.                         --
-------------------------------------------------------------------

I183: nor2
    generic map (DELAY=>1 ns, FANOUT=>1,INITDELAY=>1 fs,
            VDD=>VDD, TEMPERATURE=>TEMPERATURE)
    port map (I184_N,I189_N,I183_N);

I189: nor3
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I183_N,I1085_N,I3333_Q,I189_N);

I298: nor2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I195_N,I5555_Q,I298_N);

I9999: rctr_18stage
    generic map (q18_FANOUT=>1,testclock_fanout=>1,
            testoutput_fanout=>1,VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I298_N,I227_N,RCTR_18Q18,RCTR_18TC,RCTR_18TO);

I1401: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (RCTR_18TC, TESTCLOCK);
```

```
I296: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (RCTR_18TO,I296_N);

I295: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I227_N,I295_N);

I294: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I3333_Q,I294_N);

I1682: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I296_N,I1682_N);

I1683: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I295_N,I1683_N);

I1684: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I294_N,I1684_N);

I1093: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1682_N,TESTOUTPUT);

I1092: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1683_N,TUNERST);

I1094: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1684_N,FCHFFN);

I192: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1111_Q,I192_N);

I3333: dff_pedg_rbar
    generic map (Q_FANOUT=> 3, QN_FANOUT=> 1, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (VCC,I173_N,I169_N,I3333_Q,I3333_QN);
```

```
I193: nand2
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I192_N,I3333_QN,I193_N);

I204: nand2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I195_N,I167_N,I204_N);

I205: inv
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I204_N,I205_N);

I227: inv
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I193_N,I227_N);

I1084: cmos_in
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (FREQCH,I1084_N);

I173: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1084_N,I173_N);

I1085: cmos_in
    generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (HOLD,I1085_N);

I1086: cmos_in
    generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (HOME,I1086_N);

I1061: cmos_in
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (SETDIR,I1061_N);

I321: nand2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1086_N,I1061_N,I321_N);

I168: nor2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1085_N,I321_N,I168_N);
```

```
I169: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I168_N,I169_N);

I300: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I2222_Q,I300_N);

I1685: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I300_N,I1685_N);

I1095: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1685_N,TC1);

I167: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1086_N,I167_N);

I166: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1061_N,I166_N);

I1087: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1693_N,CB);

I1693: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I155_N,I1693_N);

I292: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I205_N,I292_N);

I1686: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I292_N,I686_N);

I1041: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I686_N,TUNECYC);
```

```
I1088: cmos_in
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (CA,I1088_N);

I151: nor2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I205_N,I1088_N,I151_N);

I152: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I151_N,I152_N);

I155: inv
    generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I152_N,I155_N);

I188: nor2
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (RCTR_18Q18,I187_N,I188_N);

I187: nor2
    generic map (DELAY=>1 ns, FANOUT=>1,INITDELAY=>1 fs,
          VDD=>VDD, TEMPERATURE=>TEMPERATURE)
    port map (I188_N,I193_N,I187_N);

I290: nor2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I188_N,I1086_N,I290_N);

I291: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I290_N,I291_N);

I293: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I291_N,I293_N);

I1687: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I293_N,I1687_N);

I1096: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
          TEMPERATURE=> TEMPERATURE)
    port map (I1687_N,NOTUNE);
```

```
I287: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I155_N,I287_N);

I1688: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I287_N,I1688_N);

I1097: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1688_N,CLOCK);

I1089: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=> 1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I1692_N,CC);

I1692: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I299_N,I1692_N);

I299: inv
    generic map (DELAY=>1 ns, FANOUT=>5, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I155_N,I299_N);

I147: nand2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I5555_Q,I4444_Q,I147_N);

I4444: dff_pedg_rbar
    generic map (Q_FANOUT=> 2, QN_FANOUT=> 1, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I147_N,I299_N,VCC,I4444_Q,open);

I5555: dff_pedg_rbar
    generic map (Q_FANOUT=> 2, QN_FANOUT=> 1, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (I4444_Q,I299_N,VCC,I5555_Q,open);

I1090: cmos_in
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (PULSEOFF,I1090_N);

I272: nor2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1090_N,RCTR_13Q13,I272_N);
```

```
I273: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I272_N,I273_N);

I8888: rctr_13stage
    generic map (q13_FANOUT=>1,VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I299_N,I138_N,RCTR_13Q13);

I7777: rctr_9stage
    generic map (q9_FANOUT=>1,VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I299_N,I139_N,RCTR_9Q9);

I1091: cmos_in
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (RSTPUL,I1091_N);

I138: inv
    generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I136_N,I138_N);

I281: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I1091_N,I281_N);

I6666: dff_pedg_rbar
    generic map (Q_FANOUT=> 2, QN_FANOUT=> 1, VDD=> VDD,
            TEMPERATURE=> TEMPERATURE)
    port map (VCC,I273_N,I281_N,I6666_Q,open);

I136: nor2
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I6666_Q,I1091_N,I136_N);

I139: nor2
    generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I6666_Q,I140_N,I139_N);

I140: nor2
    generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs,
            VDD=>VDD, TEMPERATURE=>TEMPERATURE)
    port map (I139_N,RCTR_9Q9,I140_N);

I289: nor2
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
            TEMPERATURE=>TEMPERATURE)
    port map (I140_N,I136_N,I289_N);
```

```
I283: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I289_N,I283_N);

I284: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I283_N,I284_N);

I1689: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I284_N,I1689_N);

I1098: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=> TEMPERATURE)
    port map (I1689_N,EXHOLDN);

I285: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I138_N,I285_N);

I1590: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I285_N,I1590_N);

I1099: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=> TEMPERATURE)
    port map (I1590_N,PULSES);

I301: inv
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I138_N,I301_N);

I1691: out_buff
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
    port map (I301_N,I1691_N);

I1100: cmos_pad
    generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD,
          TEMPERATURE=> TEMPERATURE)
    port map (I1691_N,PULSEI);

end struct;
```

```
use STD.Standard.all;
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

entity chp395mod is

   port  (VSWR, TUNECMD, FREQCH, HOLD, HOME:        in   bit;
          SETDIR, CA, PULSEOFF, RSTPUL:             in   bit;
          TESTCLOCK, CB, CC, TESTOUTPUT, TUNERST:   out  bit;
          FCHFFN, TC1, TUNECYC, NOTUNE, CLOCK:      out  bit;
          EXHOLDN, PULSES, PULSEI:                  out  bit);

end chp395mod;

architecture chp395 of chp395mod is

   component chp395
      generic (TEMPERATURE: real;  VDD: real);

      port  (VSWR, TUNECMD, FREQCH, HOLD, HOME:     in   bit;
             SETDIR, CA, PULSEOFF, RSTPUL:          in   bit;
             TESTCLOCK, CB, CC, TESTOUTPUT, TUNERST:  out  bit;
             FCHFFN, TC1, TUNECYC, NOTUNE, CLOCK:   out  bit;
             EXHOLDN, PULSES, PULSEI:               out  bit);
   end component;

   for NO: chp395 use entity work.chp395(struct);

begin

-------------------------------------------------------------------
-- This section passes through the Voltage VDD and temperature values to the --
-- structure which passes it through generic maps into the gates. This    --
-- method allows the user to change the values as needed without having to  --
-- change anything else in other source code programs.                    --
-------------------------------------------------------------------

      NO: chp395
       generic map (TEMPERATURE=>298.15, VDD=>10.0)
       port map   (VSWR, TUNECMD, FREQCH, HOLD, HOME,
                   SETDIR, CA, PULSEOFF, RSTPUL,
                   TESTCLOCK, CB, CC, TESTOUTPUT, TUNERST,
                   FCHFFN, TC1, TUNECYC, NOTUNE, CLOCK,
                   EXHOLDN, PULSES, PULSEI);

end chp395;
```

```
use STD.textio.all;
use STD.Standard.all;
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-----------------------------------------------------------------
-- Test bench for simulation of the SM-B-746395 integrated circuit of the --
-- PRC-70 radio.  The clock cycle is 4000 ns.                   --
-----------------------------------------------------------------

entity test_chp395 is
end test_chp395;

architecture test395 of test_chp395 is

signal VSWR          : bit:='0';
signal TUNECMD       : bit:='0';
signal FREQCH        : bit:='0';
signal HOLD          : bit:='0';
signal HOME          : bit:='1';
signal SETDIR        : bit:='1';
signal CA            : bit:='1';
signal PULSEOFF      : bit:='0';
signal RSTPUL        : bit:='1';
signal TESTCLOCK     : bit;
signal CB            : bit;
signal CC            : bit;
signal TESTOUTPUT    : bit;
signal TUNERST       : bit;
signal FCHFFN        : bit;
signal TC1           : bit;
signal TUNECYC       : bit;
signal NOTUNE        : bit;
signal CLOCK         : bit;
signal EXHOLDN       : bit;
signal PULSES        : bit;
signal PULSEI        : bit;
signal L1            : bit:='0';
signal L2            : bit:='0';
signal CLOCKA        : bit:='1';
signal CLOCKB        : bit:='0';
signal CLOCKC        : bit:='1';
signal STOP          : bit:='0';


  component chp395mod

    port   (VSWR, TUNECMD, FREQCH, HOLD, HOME:        in   bit;
            SETDIR, CA, PULSEOFF, RSTPUL:             in   bit;
            TESTCLOCK, CB, CC, TESTOUTPUT, TUNERST:   out  bit;
            FCHFFN, TC1, TUNECYC, NOTUNE, CLOCK:      out  bit;
            EXHOLDN, PULSES, PULSEI:                  out  bit);
```

```
  end component;

  for CH: chp395mod use entity work.chp395mod(chp395);

begin -- input stimulus

VSWR      <= '1' after 20000 ns;

TUNECMD   <= '1' after 18000 ns,  '0' after 22000 ns,
             '1' after 26000 ns,  '0' after 42000 ns,
             '1' after 44000 ns,  '0' after 37956000 ns,
             '1' after 37960000 ns;

FREQCH    <= '1' after 8000 ns,   '0' after 10000 ns,
             '1' after 32000 ns,  '0' after 34000 ns,
             '1' after 1610910000 ns,'0' after 1610912000 ns;

HOLD      <= '1' after 6000 ns,   '0' after 12000 ns,
             '1' after 48000 ns,  '0' after 50000 ns;

HOME      <=                     '0' after 14000 ns,
             '1' after 36000 ns,  '0' after 44108000 ns,
             '1' after 1610906000 ns,'0' after 1610908000 ns;

SETDIR    <= '0' after 24000 ns,  '1' after 28000 ns,
             '0' after 30000 ns,  '1' after 38000 ns,
             '0' after 40000 ns,  '1' after 46000 ns,
             '0' after 37958000 ns, '1' after 37962000 ns;

PULSEOFF  <=                     '1' after 1078000 ns,
             '0' after 18494000 ns;

RSTPUL    <= '0' after 1076000 ns, '1' after 1080000 ns,
             '0' after 2106000 ns, '1' after 18492000 ns,
             '0' after 18496000 ns, '1' after 37954000 ns;

L1        <= '1' after 1082000 ns, '0' after 2100000 ns,
             '1' after 18498000 ns, '0' after 37952000 ns,
             '1' after 37964000 ns, '0' after 44102000 ns;

L2        <= '1' after 52000 ns, '0' after 1074000 ns,
             '1' after 2108000 ns, '0' after 18490000 ns,
             '1' after 44106000 ns, '0' after 1610904000 ns;

CLOCKA    <= '0' after 2000 ns,   '1' after 4000 ns,
             '0' after 16000 ns,  '1' after 52000 ns,
             '0' after 1074000 ns,  '1' after 1082000 ns,
             '0' after 2100000 ns,  '1' after 2102000 ns,
             '0' after 2104000 ns,  '1' after 2108000 ns,
             '0' after 18490000 ns, '1' after 18498000 ns,
             '0' after 37952000 ns, '1' after 44102000 ns,
             '0' after 44106000 ns, '1' after 1610904000 ns;

CLOCKB    <= not CLOCKB after 2000 ns;
```

```
CLOCKC     <= not CLOCKC after 2000 ns;
STOP       <= '1' after 1610914000 ns;

CH: chp395mod
port map (VSWR, TUNECMD, FREQCH, HOLD, HOME,
        SETDIR, CA, PULSEOFF, RSTPUL,
        TESTCLOCK, CB, CC, TESTOUTPUT, TUNERST,
        FCHFFN, TC1, TUNECYC, NOTUNE, CLOCK,
        EXHOLDN, PULSES, PULSEI);

P1: process -- process to incorporate a perpetual running clock into the test
   begin

     wait for 1 fs;
     if L1 = '1' then
       CA <= CLOCKB;
     elsif L2 = '1' then
       CA <= CLOCKC;
     else
       CA <= CI OCKA;
     end if;

     wait on L1, L2, CLOCKA,CLOCKB,CLOCKC;

   end process P1;

-----------------------------------------------------------------
-- This next process is a specially designed process based on work done by --
-- Georgia Tech University to abbreviate the simulation report because it  --
-- required too much space in memory to report all output signals at all   --
-- time intervals. Specific lines representing events in the test were taken --
-- from a physical test sweep of the chip and put into an ASCII file to be --
-- used as a control for the VHDL simulation. The simulation outputs at --
-- selected times are compared to the ASCII file. Mismatches and where  --
-- they occurred are reported along with the outputs at the selected times. --
-- The selection of time depends on the line number of the vector given in --
-- the physical test sweep ASCII file.                                  --
-----------------------------------------------------------------

P2: process

   file INPUT_FILE :      text is in "Physical_test_file.dat";

-- the ASCII file of physical test sweep

   file OUTPUT_FILE:      text is out "Compare_data.dat";

-- the output ASCII file which reports simulation output and mismatches with
-- test.

   variable Johns_line:   line;
   variable My_line:      line;
   variable Scribe_line:  line;
   variable Johns_input:  string(1 to 13);
```

```
  variable Johns_signals: string(1 to 13);
  variable My_signals:   string(1 to 11);
  variable Johns_bit:    bit;
  variable My_bit:       bit;
  variable Johns_number: integer;
  variable The_time:     time;
  variable Johns_time:   time;
  variable My_time:      time;
  variable eod:          Boolean:=false;
  variable err:          string(1 to 5);
  variable show_signal:  string(1 to 11):="0123456789A";
  variable bad_signals:  string(1 to 11);
  variable Z:            character:='Z';
  variable old_time:     time:=0Us;
  variable nine_signals: string(1 to 11);
  variable j:            integer;

 begin

 wait for 1.8 us;

 L1: loop

-- Sections or columns from ASCII file are put into string variables.

    bad_signals := "          ";
    Johns_line  := new STRING'("");
    My_line     := new STRING'("");
    Scribe_line := new STRING'("");
    READLINE(INPUT_FILE,Johns_line);
    eod := endfile(INPUT_FILE);
    READ(Johns_line,Johns_number);
    READ(Johns_line,Johns_input);
    READ(Johns_line,Johns_signals);
    Johns_time:= (2us * (Johns_number-1)) - old_time; -- time of an event

    j:=0;
    L2: for i in 1 to 13 loop

        if i /= 9 and i /= 13 then
          j:= j + 1;
          nine_signals(j):=johns_signals(i);
        end if;
      end loop L2;

    wait for Johns_time;
    The_time:= now;

-- signal outputs of simulation are put into single string variable.

    if testclock ='0' then My_signals(1):= '0';
                 else My_signals(1):= '1';
    end if;
    if testoutput ='0' then My_signals(2):= '0';
```

```
            else My_signals(2):= '1';
end if;
if notune ='0' then My_signals(3):= '0';
          else My_signals(3):= '1';
end if;
if tunerst ='0' then My_signals(4):= '0';
           else My_signals(4):= '1';
end if;
if tc1 ='0' then My_signals(5):= '0';
        else My_signals(5):= '1';
end if;
if tunecyc ='0' then My_signals(6):= '0';
           else My_signals(6):= '1';
end if;
if fchffn ='0' then My_signals(7):= '0';
          else My_signals(7):= '1';
end if;
if cc ='0' then My_signals(8):= '0';
       else My_signals(8):= '1';
end if;
if cb ='0' then My_signals(9):= '0';
       else My_signals(9):= '1';
end if;

if exholdn ='0' then My_signals(10):= '0';
           else My_signals(10):= '1';
end if;

if pulses ='0' then My_signals(11):= '0';
          else My_signals(11):= '1';
end if;
err:="    ";

--Compare loop: compares simulation output to physical test sweep.

   L3: for i in 1 to 11 loop
       if nine_signals(i) /= My_signals(i) and
         nine_signals(i) /= Z then
           err:="**** ";
           bad_signals(i):=show_signal(i);
       end if;
   end loop L3;

 old_time:= 2us * (Johns_number - 1);
 Write(Scribe_line,Johns_number,Justified=>RIght,Field=>9);
 Write(Scribe_line,Johns_signals,Justified=>Right,Field=>14);
 Write(Scribe_line,nine_signals,Justified=>Right,Field=>13);
 Write(Scribe_line,My_signals,Justified=>Right,Field=>13);
 Write(Scribe_line,The_time,Justified=>Right,Field=>15);
 Write(Scribe_line,err,Justified=>Right,Field=>8);
 Write(Scribe_line,bad_signals);
 WRITELINE(OUTPUT_FILE,Scribe_line);

exit L1 when eod = True;
```

```
end loop L1;

   end process P2;


P3: process  -- process to stop the simulation at a specified time.
   begin

    wait until STOP = '1';
    assert STOP ='0'
      report "end of simulation after 2 ms"
      severity FAILURE;

   end process P3;


end test395;

-- ASCII input file from physical test lab --

    1  101110000 ZZ00Z11110Z00
    2  100110000 ZZ00Z11001Z00
    3  101110000 ZZ00Z11110Z00
    4  101111000 ZZ00111110Z00
    5  101111100 ZZ00111110Z00
    6  101111000 0101110110Z00
    7  101110000 0100111110Z00
    8  101100000 0100101110Z00
    9  100100000 0100101110Z00
   10  100100010 0100101110Z00
   11  100100011 0100101110Z00
   12  100100001 0100101110Z00
   13  100000001 0100101110Z00
   14  100000011 0101011001Z00
   15  100100011 0100011001Z00
   16  100000011 0100011001Z00
   17  100000111 0100011001Z00
   18  100000011 0101100110Z00
   19  100010011 0101110001Z00
   20  100110011 0100111001Z00
   21  100010011 0100111001Z00
   22  100010001 0100111001Z00
   23  100010011 0101011001Z00
   24  100110011 0100011001Z00
   25  100111011 0100111001Z00
   26  100110011 0100111001Z00
  538  100110011 0100111001100
  539  000110011 0100111001011
  540  010110011 0100111001000
  541  110110011 0100111001000
 1051  110110011 0100111001000
 1052  111110011 0100111110100
 1053  110110011 0100111001100
```

```
 1054  010110011 0100111001011
 9246  010110011 0100111001011
 9247  110110011 0100111001100
 9248  100110011 0100111001100
 9249  000110011 0100111001011
17439  000110011 0100111001011
17440  001110011 0100111110000
17441  000110011 0100111001000
17953  000110011 0100111001100
18465  000110011 0100111001000
18977  000110011 0100111001100
18978  100110011 0100111001100
18979  100110001 0100111001100
18980  100010001 0100111001100
18981  100010011 0101011001100
18982  100110011 0100011001100
20512  101110011 0100011110100
20513  100110011 0100011001100
20514  101110011 0100011110100
20515  100110011 0100011001100
20516  101110011 1100011110100
20517  100110011 1100011001100
20518  101110011 1100011110100
22047  100110011 1100011001100
22048  101110011 1100011110100
22049  100110011 1100011001100
22050  101110011 1100011110100
22051  100110011 1100011001100
22052  101110011 0000011110100
22053  101100011 0000011110100
805373 101100011 1000011110100
805453 101100011 0110011110100
805454 101110011 0100011110100
805455 101100011 0110011110100
805456 101100111 0110011110100
805457 101100011 0101100110100
```

-- ASCII output file for comparison of simulation and physical test data --

```
 1 ZZ00Z11110Z00 ZZ00Z1110Z0 01000111010     1800 NS
 2 ZZ00Z11001Z00 ZZ00Z1101Z0 01000110110     3800 NS
 3 ZZ00Z11110Z00 ZZ00Z1110Z0 01000111010     5800 NS
 4 ZZ00111110Z00 ZZ0011110Z0 01001111010     7800 NS
 5 ZZ00111110Z00 ZZ0011110Z0 01001111010     9800 NS
 6 0101110110Z00 010111010Z0 01011101010    11800 NS
 7 0100111110Z00 010011110Z0 01001111010    13800 NS
 8 0100101110Z00 01001011OZ0 01001011010    15800 NS
 9 0100101110Z00 01001011OZ0 01001011010    17800 NS
10 0100101110Z00 01001011OZ0 01001011010    19800 NS
11 0100101110Z00 01001011OZ0 01001011010    21800 NS
12 0100101110Z00 01001011OZ0 01001011010    23800 NS
13 0100101110Z00 01001011OZ0 01001011010    25800 NS
14 0101011001Z00 01010110lZ0 01010110110    27800 NS
15 0100011001Z00 01000110lZ0 01000110110    29800 NS
```

16 0100011001Z00 01000110120 01000110110 31800 NS
17 0100011001Z00 01000110120 01000110110 33800 NS
18 0101100110Z00 01011001020 01011001010 35800 NS
19 0101110001Z00 01011100120 01011100110 37800 NS
20 0100111001Z00 01001110120 01001110110 39800 NS
21 0100111001Z00 01001110120 01001110110 41800 NS
22 0100111001Z00 01001110120 01001110110 43800 NS
23 0101011001Z00 01010110120 01010110110 45800 NS
24 0100011001Z00 01000110120 01000110110 47800 NS
25 0100111001Z00 01001110120 01001110110 49800 NS
26 0100111001Z00 01001110120 01001110110 51800 NS
538 0100111001100 01001110110 01001110110 1075800 NS
539 0100111001011 01001110101 01001110101 1077800 NS
540 0100111001000 01001110100 01001110100 1079800 NS
541 0100111001000 01001110100 01001110100 1081800 NS
1051 0100111001000 01001110100 01001110100 2101800 NS
1052 0100111110100 01001111010 01001111010 2103800 NS
1053 0100111001100 01001110110 01001110110 2105800 NS
1054 0100111001011 01001110101 01001110101 2107800 NS
9246 0100111001011 01001110101 01001110101 18491800 NS
9247 0100111001100 01001110110 01001110110 18493800 NS
9248 0100111001100 01001110110 01001110110 18495800 NS
9249 0100111001011 01001110101 01001110101 18497800 NS
17439 0100111001011 01001110101 01001110101 34877800 NS
17440 0100111110000 01001111000 01001111000 34879800 NS
17441 0100111001000 01001110100 01001110100 34881800 NS
17953 0100111001100 01001110110 01001110110 35905800 NS
18465 0100111001000 01001110100 01001110100 36929800 NS
18977 0100111001100 01001110110 01001110110 37953800 NS
18978 0100111001100 01001110110 01001110110 37955800 NS
18979 0100111001100 01001110110 01001110110 37957800 NS
18980 0100111001100 01001110110 01001110110 37959800 NS
18981 0101011001100 01010110110 01010110110 37961800 NS
18982 0100011001100 01000110110 01000110110 37963800 NS
20512 0100011110100 01000111010 01000111010 41023800 NS
20513 0100011001100 01000110110 01000110110 41025800 NS
20514 0100011110100 01000111010 01000111010 41027800 NS
20515 0100011001100 01000110110 01000110110 41029800 NS
20516 1100011110100 11000111010 11000111010 41031800 NS
20517 1100011001100 11000110110 11000110110 41033800 NS
20518 1100011110100 11000111010 11000111010 41035800 NS
22047 1100011001100 11000110110 11000110110 44093800 NS
22048 1100011110100 11000111010 11000111010 44095800 NS
22049 1100011001100 11000110110 11000110110 44097800 NS
22050 1100011110100 11000111010 11000111010 44099800 NS
22051 1100011001100 11000110110 11000110110 44101800 NS
22052 0000011110100 00000111010 00000111010 44103800 NS
22053 0000011110100 00000111010 00000111010 44105800 NS
805373 1000011110100 10000111010 10000111010 1610745800 NS
805453 0110011110100 01100111010 01100111010 1610905800 NS
805454 0100011110100 01000111010 01000111010 1610907800 NS
805455 0110011110100 01100111010 01100111010 1610909800 NS
805456 0110011110100 01100111010 01100111010 1610911800 NS
805457 0101100110100 01011001010 01011001010 1610913800 NS

```
use STD.Standard.all;
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

entity CHP396 is

  generic (TEMPERATURE, VDD: real);

  port (CLOCK:    in   bit;
        RSTSW:    in   bit;
        SWHOLDN:  in   bit;
        EXHOLDN:  in   bit;
        PCHFFN:   in   bit;
        LLIMN:    in   bit;
        RLT:      in   bit;
        VHFHI:    in   bit;
        RGT:      in   bit;
        PLT:      in   bit;
        PGT:      in   bit;
        PMAX:     in   bit;
        TC1:      in   bit;
        TUNERST:  in   bit;
        NOTUNE:   in   bit;
        SWENAN:   out  bit;
        FINE:     out  bit;
        HOME:     out  bit;
        TUNE:     out  bit;
        RLEGFF:   out  bit;
        NORMAL:   out  bit;
        RSTPUL:   out  bit;
        EXEC:     out  bit;
        ENPCTR:   out  bit;
        LCMD:     out  bit;
        MODSAMPN: out  bit;
        ENPSIG:   out  bit;
        RMINUSN:  out  bit;
        RPLUSN:   out  bit;
        SETDIR:   out  bit;
        PULSEDIR: out  bit;
        DIR:      out  bit;
        CRSE:     out  bit;
        EXDISN:   out  bit;
        PRCTR:    out  bit;
        HOLD:     out  bit);

end CHP396;

architecture struct of CHP396 is

-------------------------------------------------------------------

-- All components obtained from GEM gate behavioral descriptions written by --
-- Ken Keyes. Each component gets its delay values from a table and packages--
-- which calculate delay values based on fanout, VDD, and temperature passed --
```

```
-- in through generic maps.                                    --
-----------------------------------------------------------------

  component cmos_in
    generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          CMOS_IN_TABLE:cell_values:=work.tables.cmos_in_table);
    port (INPUT:  in  bit;
          OUTPUT: out bit);
  end component;

  component out_buff
    generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          OUT_BUFF_TABLE:cell_values:=work.tables.out_buff_table);
    port (INPUT:  in  bit;
          OUTPUT: out bit);
  end component;

  component cmos_pad
    generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          CMOS_PAD_TABLE:cell_values:=work.tables.cmos_pad_table);
    port (INPUT:  in  bit;
          OUTPUT: out bit);
  end component;

  component inv
    generic (DELAY:time;FANOUT:integer;VDD:real;
          TEMPERATURE:real;
          INV_TABLE:cell_values:=work.tables.inv_table);
    port (INPUT:  in  bit;
          OUTPUT: out bit);
  end component;

  component nand2
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0 fs;
          VDD:real;TEMPERATURE:real;
          NAND2_TABLE:cell_values:=work.tables.nand2_table);
    port (INPUT1: in  bit;
          INPUT2: in  bit;
          OUTPUT: out bit);
  end component;

  component nand3
    generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0 fs;
          VDD:real;TEMPERATURE:real;
          NAND3_TABLE:cell_values:=work.tables.nand3_table);
    port (INPUT1: in  bit;
          INPUT2: in  bit;
          INPUT3: in  bit;
          OUTPUT: out bit);
  end component;
```

```
component nand4
  generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0 fs;
        VDD:real;TEMPERATURE:real;
        NAND4_TABLE:cell_values:=work.tables.nand4_table);
  port (INPUT1:  in   bit;
      INPUT2:  in   bit;
      INPUT3:  in   bit;
      INPUT4:  in   bit;
      OUTPUT:  out  bit);
end component;

component nor2
  generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0 fs;
        VDD:real;TEMPERATURE:real;
        NOR2_TABLE:cell_values:=work.tables.nor2_table);
  port (INPUT1:  in   bit;
      INPUT2:  in   bit;
      OUTPUT:  out  bit);
end component;

component nor3
  generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0 fs;
        VDD:real;TEMPERATURE:real;
        NOR3_TABLE:cell_values:=work.tables.nor3_table);
  port (INPUT1:  in   bit;
      INPUT2:  in   bit;
      INPUT3:  in   bit;
      OUTPUT:  out  bit);
end component;

component nor4
  generic (DELAY:time;FANOUT:integer;INITDELAY:time:=0 fs;
        VDD:real;TEMPERATURE:real;
        NOR4_TABLE:cell_values:=work.tables.nor4_table);
  port (INPUT1:  in   bit;
      INPUT2:  in   bit;
      INPUT3:  in   bit;
      INPUT4:  in   bit;
      OUTPUT:  out  bit);
end component;

component exor
  generic (DELAY:time;FANOUT:integer;VDD:real;
        TEMPERATURE:real;
        EXOR_TABLE:cell_values:=work.tables.exor_table);
  port (INPUT1:  in   bit;
      INPUT2:  in   bit;
      OUTPUT:  out  bit);
end component;

component dff_pedg_rbar
  generic (Q_FANOUT:integer; QN_FANOUT:integer;
        VDD:real; TEMPERATURE:real);
  port (D:     in   bit;
```

```
    CLK:   in  bit;
    RESET: in  bit;
    Q:     out bit;
    QBAR:  out bit);
end component;

component dff_pedg_srber
  generic (Q_FANOUT:integer; QN_FANOUT:integer;
      VDD:real; TEMPERATURE:real);
  port (D:    in  bit;
    CLK:   in  bit;
    RESET: in  bit;
    SET:   in  bit;
    Q:     out bit;
    QBAR:  out bit);
end component;

-----------------------------------------------------------
-- All signal names correspond to the MIF file signal names (except for the --
-- $ sign which is a reserved character in VHDL) for easy cross reference  --
-- with the design drawings which were produced from a mentor graphics     --
-- workstation.                                            --
-----------------------------------------------------------

signal VCC: bit:= '1';
signal gnd: bit:= '0';

signal N17,N102,N107,N129,N132,N143,N144,N148,N154,N155,N156,N157,N3: bit;
signal N160,N168,N173,N174,N180,N184,N190,N196,N2,N29,N211,N214,N34: bit;
signal N953,N955,N959,N961,N968,N974,N976,N979,N983,N988,N995,N997: bit;
signal N1001,N1004,N1008,N1013,N1017,N1019,N1020,N1046,N1053,N1056: bit;
signal N1063,N1150,N1154,N1159,N1161,N1175,N1181,N1184,N1196,N1198: bit;
signal N1201,N1221,N1226,N1234,N1236,N1266,N1285,N1293,N945,N946,N948: bit;
signal N1301,N1303,N1313,N1316,N1325,N1329,N1341,N1343,N1344,N1348: bit;
signal N1354,N1356,N1359,N1362,N1381,N1389,N1394,N1398,N2845,N2846: bit;
signal N1400,N1404,N1405,N1415,N1417,N1418,N1420,N1431,N1442,N1446: bit;
signal N1447,N1449,N1452,N1458,N1468,N1469,N1471,N1472,N1474,N1478: bit;
signal N1489,N1495,N1496,N1502,N1504,N1506,N1509,N1516,N1533,N1534: bit;
signal N1539,N1544,N1545,N1548,N1550,N1559,N1562,N1572,N1581,N1582: bit;
signal N1601,N1602,N1722,N1725,N1733,N1748,N1750,N1756,N1763,N1767: bit;
signal N1778,N1780,N1781,N1793,N1797,N1798,N1799,N1800,N1801,N1803: bit;
signal N1881,N1887,N1894,N1895,N1897,N1899,N1900,N1902,N1903,N1907: bit;
signal N1908,N1911,N1914,N1916,N1920,N1922,N1934,N1940,N1941,N1943: bit;
signal N1946,N1948,N1950,N1951,N1953,N1961,N1962,N1964,N1966,N1968: bit;
signal N1973,N1974,N1977,N1984,N1987,N1989,N1991,N1992,N1997,N2001: bit;
signal N2003,N2005,N2008,N2009,N2013,N2201,N2401,N2407,N2426,N2438: bit;
signal N2439,N2442,N2443,N2467,N2469,N2472,N2474,N2475,N2477,N2479: bit;
signal N2481,N2483,N2484,N2485,N2487,N2489,N2493,N2494,N2497,N2499: bit;
signal N2500,N2502,N2507,N2511,N2515,N2517,N2518,N2521,N2843,N2844: bit;
signal N2847,N2848,N2849,N2850,N2851,N2852,N2853,N2854,N2855,N2856: bit;
signal N2857,N2858,N2860,N2861,N2862,N2863,N2864,N2865,N2866,N2867: bit;

for all:   cmos_in use entity work.cmos_in(behavioral);
for all:   cmos_pad use entity work.cmos_pad(behavioral);
```

```
  for all:   out_buff use entity work.out_buff(behavioral);
  for all:   inv use entity work.inv(behavioral);
  for all:   nand2 use entity work.nand2(behavioral);
  for all:   nand3 use entity work.nand3(behavioral);
  for all:   nand4 use entity work.nand4(behavioral);
  for all:   nor2 use entity work.nor2(behavioral);
  for all:   nor3 use entity work.nor3(behavioral);
  for all:   nor4 use entity work.nor4(behavioral);
  for all:   exor use entity work.exor(behavioral);
  for all:   dff_pedg_rbar use entity work.dff_pedg_rbar(struct);
  for all:   dff_pedg_srbar use entity work.dff_pedg_srbar(struct);

begin

  I1301: cmos_in
     generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (CLOCK,N2401);

  I1302: cmos_pad
     generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (N2860,SWENAN);

  I1304: cmos_in
     generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (RSTSW,N2407);

  I1307: cmos_in
     generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (SWHOLDN,N1417);

  I1309: cmos_in
     generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (EXHOLDN,N1418);

  I1311: cmos_in
     generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (FCHFFN,N1803);

  I1313: cmos_pad
     generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (N2861,FINE);

  I1314: cmos_pad
     generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (N2866,HOME);

  I1316: cmos_in
     generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (LLIMN,N1881);

  I1320: cmos_pad
     generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
     port map (N2438,TUNE);
```

```
I1321: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2864,RLEGFF);

I1322: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2442,NORMAL);

I1323: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (RLT,N2443);

I1328: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (VHFHI,N1458);

I1329: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>6, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (RGT,N1763);

I1332: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (PLT,N1539);

I1333: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (PGT,N1722);

I1334: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2469,RSTPUL);

I1335: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (PMAX,N1781);

I1339: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N945,N2467);

I1341: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2467,N2851);

I1344: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N948,N2472);

I1345: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2850,EXEC);

I1347: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
```

```
   port map (N953,N2475);

I1348: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2475,N2849);

I1349: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2474,ENPCTR);

I1351: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N34,N2477);

I1352: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2848,LCMD);

I1354: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N156,N2847);

I1355: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2479,MODSAMPN);

I1358: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2481,N2484);

I1359: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2484,N2483);

I1360: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2846,ENPSIG);

I1362: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2485,N2845);

I1363: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2487,RMINUSN);

I1365: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N211,N2489);

I1366: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2844,RPLUSN);
```

```
I1369: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N154,N2494);

I1370: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2494,N2843);

I1371: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2493,SETDIR);

I1373: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (TC1,N1780);

I1374: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2497,N2499);

I1375: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2853,PULSEDIR);

I1377: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2500,N2852);

I1378: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2502,DIR);

I1382: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N995,N2507);

I1383: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2867,CRSE);

I1385: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N968,N2511);

I1386: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2511,N2855);

I1387: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2854,EXDISN);

I1389: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
```

```
   port map (TUNERST,N1992);

I1391: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2515,N2518);

I1392: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2518,N2857);

I1393: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2517,PRCTR);

I1395: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1301,N2521);

I1396: cmos_pad
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2856,HOLD);

I1399: cmos_in
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (NOTUNE,N1778);

I1701: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2855,N2854);

I1702: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2851,N2469);

I1703: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2472,N2850);

I1704: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2849,N2474);

I1705: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2477,N2848);

I1706: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2847,N2479);

I1707: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2483,N2846);
```

```
I1708: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2845,N2487);

I1709: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2489,N2844);

I1710: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2843,N2493);

I1711: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2499,N2853);

I1712: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2852,N2502);

I1713: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2857,N2517);

I1714: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2521,N2[illegible]6);

I1715: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2858,N2860);

I1716: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2862,N2861);

I1717: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2426,N2866);

I1718: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2865,N2438);

I1719: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2439,N2864);

I1720: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2863,N2442);

I1721: out_buff
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
```

```
   port map (N2507,N2867);

I1201: exor
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1974,N1767,N1973);

I1162: exor
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N157,N196,N214);

I1161: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1431,N1562,N1799);

I1141: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1181,N1175,N1198);

I915: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N154,N2013);

I913: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1017,N2009);

I912: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1934,N2008);

I911: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2001,N2003);

I909: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1161,N1997);

I907: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1992,N1991);

I906: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1987,N1989);

I905: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1966,N1977);

I902: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1968,N1973,N1966);
```

```
I901: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1962,N1964);

I892: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1948,N1951);

I891: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1478,N1946);

I890: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1940,N1941);

I889: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1922,N1797);

I888: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1914,N1916);

I887: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1920,N1911);

I886: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1908,N1907);

I885: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port m  N1903,N1798);

I884: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1303,N1572,N1903);

I883: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1899,N1897);

I882: inv
   ge  p (DE  1 ns, FANOUT=>4, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1894,N1895);

I811: dff_pedg_srbar
   generic map (Q_FANOUT=> 9, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERAT  > TEMPERATURE)
   port map (N1800,N  ,N1907,N1801,N1420,open);
```

```
I810: inv
   generic map (DELAY=>1 ns, FANOUT=>9, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1420,N2005);

I786: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1950,N1953);

I784: nand3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1478,N1940,N983,N1987);

I783: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1495,N1506,N1733);

I782: dff_pedg_sbar
    generic map (Q_FANOUT=> 3, QN_FANOUT=> 1, VDD=> VDD,
           TEMPERATURE=> TEMPERATURE)
    port map (N1941,N1951,N1943,N1478,N1725,open);

I781: nor4
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1468,N1420,N1472,N1756,N1478);

I779: nor4
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1934,N1799,N1601,N1415,N1920);

I653: nand3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1159,N1803,N1431,N1161);

I652: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1582,N1502,N1581);

-----------------------------------------------------------------
-- Some gate instantiations required an initial delay because they were paired-
-- with other gates to form a Flip-Flop pattern that would have an undefined --
-- state when simulation began. To offset the race condition in the feedback --
-- loop one of the gates in the pattern was given a 1 fs delay to wait until --
-- the other had received and responded to its input.                        --
-----------------------------------------------------------------

I651: nor3
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
          TEMPERATURE=>TEMPERATURE)
   port map (N1581,N1756,N1763,N1502);

I645: nand4
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1962,N1533,N1516,N1559,N1562);
```

```
I643: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1550,N1458,N1559);

I642: nand2
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N1545,N1756,N1550);

I641: nand3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1548,N1544,N1550,N1545);

I640: nand3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1356,N1756,N1763,N1544);

I639: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1539,N1722,N983);

I638: nand3
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1950,N1763,N1767,N1534);

I637: nand3
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1489,N1534,N1763,N1533);

I635: dff_pedg_rbar
   generic map (Q_FANOUT=> 2, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1733,N1961,N1950,N2500,open);

I633: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N1582,N1748,N1516);

I632: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1516,N1504,N1748);

I630: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1962,N1940,N1509);

I629: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1767,N1533,N1506);

I628: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1509,N1502,N1504);
```

```
I627: nor2
   generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1496,N1469,N1940);

I626: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1533,N1469);

I625: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1495,N1496);

I623: nand2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2443,N1763,N1489);

I622: nand3
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2443,N1489,N1534,N1495);

I620: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1946,N1961,N1948);

I618: nand2
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1472,N1474,N1750);

I617: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1471,N1469,N1442);

I616: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1471,N1452);

I615: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1953,N2862);

I614: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2005,N2426);

I613: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1431,N2863);

I612: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1452,N2865);
```

```
I611: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1725,N1449);

I610: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1449,N2439);

I609: nand3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1756,N1767,N1446,N1447);

I608: nand3
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N107,N1442,N1750,N1601);

I607: nand2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1447,N1602,N1415);

I604: inv
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1756,N1431);

I601: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N1471,N1446,N1548);

I600: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1548,N1420,N1446);

I599: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1234,N1881,N1800);

I598: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1756,N1950,N1234);

I597: nand3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2401,N1417,N1418,N1894);

I594: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1911,N1800,N1914);

I591: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1400,N1293,N1404);
```

```
I590: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N1404,N1341,N1293);

I589: nand2
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1803,N1325,N1303);

I588: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1404,N1582,N1325);

I587: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1329,N1400);

I586: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1420,N1887,N1329);

I585: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1398,N2858);

I584: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1394,N2407,N1390);

I582: inv
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2401,N2201);

I581: nand3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1221,N1973,N1984,N1381);

I567: nor3
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1420,N1725,N1992,N1150);

I566: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1572,N1362,N1389);

I565: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N1348,N1922,N1362);

I564: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1471,N1201,N1602);
```

```
I563: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1881,N1750,N1359);

I561: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1881,N1962,N1201);

I560: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1881,N1748,N1474);

I559: nor3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1953,N1343,N1344,N1341);

I553: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N17,N2);

I552: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1303,N1801);

I551: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1301,N1801,N2001);

I549: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1582,N1293,N1394);

I547: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N17,N1285,N1899);

I546: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1313,N1316,N1285);

I545: dff_pedg_rbar
   generic map (Q_FANOUT=> 3, QN_FANOUT=> 3, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1266,N1895,N1301,N1793,N1313);

I544: dff_pedg_rbar
   generic map (Q_FANOUT=> 4, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1793,N1895,N1301,N17,open);

I543: dff_pedg_rbar
   generic map (Q_FANOUT=> 3, QN_FANOUT=> 3, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1897,N1895,N1301,N1900,N1316);
```

```
I542: dff_pedg_rbar
   generic map (Q_FANOUT=> 2, QN_FANOUT=> 3, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1900,N1895,N1301,N1902,N1266);

I541: dff_pedg_rbar
   generic map (Q_FANOUT=> 8, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1601,N1572,N2005,N1756,open);

I540: dff_pedg_rbar
   generic map (Q_FANOUT=> 5, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1799,N1572,N2005,N1471,N1943);

I539: dff_pedg_rbar
   generic map (Q_FANOUT=> 6, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1415,N1572,N2005,N1950,open);

I537: dff_pedg_rbar
   generic map (Q_FANOUT=> 8, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1916,N1405,N1797,N1572,open);

I535: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1750,N1226);

I533: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1964,N1405,VCC,N1221,N1974);

I532: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1961,N1968);

I529: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1908,N1198,N1196);

I528: nor3
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1196,N1303,N1405,N1908);

I526: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1175,N1961,N1420,N1184,open);

I525: dff_pedg_rbar
   generic map (Q_FANOUT=> 2, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
```

```
   port map (N1184,N1961,N1420,N1181,open);

I524: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1181,N1961,N1420,open,N1175);

I521: inv
   generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1472,N1159);

I482: nor3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1756,N1950,N1471,N1354);

I475: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1359,N1362,N1348);

I471: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1063,N1977,N1150,open,N1063);

I470: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1989,N1961,N1991,N1056,open);

I469: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1053,N1063,N1150,N1984,N1053);

I468: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1381,N1046);

I467: nor3
   generic map (DELAY=>1 ns, FANOUT=>6, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1056,N1046,N1778,N1301);

I466: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2005,N1389,N2515);

I465: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1019,N1226,N1020);

I464: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N1922,N1020,N1019);
```

```
I463: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1008,N1013,N1017);

I462: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1405,N1019,N1013);

I461: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N154,N2201,N1008);

I459: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N129,N1004);

I458: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N2008,N1572,N2009,N1001,open);

I457: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1472,N997,N995);

I456: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N988,N1922,N2013,N997,open);

I455: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (VCC,N1159,N2013,N988,open);

I454: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N154,N974,N979);

I453: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1472,N1344,N976);

I452: nor3
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N979,N976,N1004,N974);

I451: nor2
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N974,N1001,N968);

I450: nor2
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
```

```
   port map (N1781,N955,N2481);

I449: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N959,N961);

I448: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1159,N968,N959);

I447: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1344,N955);

I445: nor3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N968,N34,N1354,N953);

I444: nor3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N961,N948,N1420,N946);

I443: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N955,N946,N945);

I50: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N214,N2497);

I49: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2481,N2497,N211);

I48: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2481,N214,N2485);

I44: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2500,N196);

I43: inv
   generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1962,N1767);

I42: nand2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N156,N1940,N190);

I41: nand2
   generic map (DELAY=>1 ns, FANOUT=>2, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N160,N184,N180);
```

```
I40: nand4
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N180,N2005,N144,N190,N184);

I39: nand2
   generic map (DELAY=>1 ns, FANOUT=>5, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N174,N1722,N1962);

I38: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N180,N168,N174);

I37: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1722,N173);

I36: nand2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N156,N173,N144);

I35: nand4
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N168,N1356,N1763,N1722,N160);

I34: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2407,N132,N157);

I33: inv
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N155,N156);

I32: nand3
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2485,N1725,N1950,N148);

I31: nand4
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N168,N2005,N144,N190,N143);

I30: nand2
   generic map (DELAY=>1 ns, FANOUT=>3, INITDELAY=>1 fs, VDD=>VDD,
         TEMPERATURE=>TEMPERATURE)
   port map (N148,N143,N168);

I29: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N3,N154,N132);

I27: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2201,N1572,N129);
```

```
I23: dff_pedg_rbar
   generic map (Q_FANOUT=> 5, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (N1934,N1572,N2005,N1472,open);

I22: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1343,N102,N107);

I21: nor2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1236,N1780,N102);

I18: nor3
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1780,N1236,N1343,N1934);

I17: dff_pedg_rbar
   generic map (Q_FANOUT=> 1, QN_FANOUT=> 2, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (VCC,N2005,N1997,N1468,N1236);

I16: dff_pedg_rbar
   generic map (Q_FANOUT=> 3, QN_FANOUT=> 1, VDD=> VDD,
         TEMPERATURE=> TEMPERATURE)
   port map (VCC,N1159,N2003,N1343,open);

I13: nand2
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1961,N2201,N155);

I12: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1922,N34);

I11: inv
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1887,N948);

I10: inv
   generic map (DELAY=>1 ns, FANOUT=>1, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2407,N29);

I9: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1793,N1902,N1356);

I8: nor2
   generic map (DELAY=>1 ns, FANOUT=>8, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N17,N1313,N1961);

I7: nand2
   generic map (DELAY=>1 ns, FANOUT=>3, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N29,N3,N1344);
```

```
I6: nor2
   generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1316,N17,N1582);

I5: nor2
   generic map (DELAY=>1 ns, FANOUT=>5, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1266,N1313,N154);

I4: nor2
   generic map (DELAY=>1 ns, FANOUT=>5, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1902,N1316,N1922);

I3: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1900,N1266,N1887);

I2: nor2
   generic map (DELAY=>1 ns, FANOUT=>4, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N2,N1900,N1405);

I1: nor2
   generic map (DELAY=>1 ns, FANOUT=>2, VDD=>VDD, TEMPERATURE=>TEMPERATURE)
   port map (N1793,N2,N3);

end struct;


use STD.Standard.all;
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

entity chp396mod is

  port (CLOCK:   in     bit;
      RSTSW:   in     bit;
      SWHOLDN: in     bit;
      EXHOLDN: in     bit;
      FCHFFN:  in     bit;
      LLIMN:   in     bit;
      RLT:     in     bit;
      VHFHI:   in     bit;
      RGT:     in     bit;
      PLT:     in     bit;
      PGT:     in     bit;
      PMAX:    in     bit;
      TC1:     in     bit;
      TUNERST: in     bit;
      NOTUNE:  in     bit;
      SWENAN:  out    bit;
      FINE:    out    bit;
      HOME:    out    bit;
      TUNE:    out    bit;
```

```
        RLEGFF: out     bit;
        NORMAL: out     bit;
        RSTPUL: out     bit;
        EXEC:   out     bit;
        ENPCTR: out     bit;
        LCMD:   out     bit;
        MODSAMPN: out   bit;
        ENPSIG: out     bit;
        RMINUSN: out    bit;
        RPLUSN: out     bit;
        SETDIR: out     bit;
        PULSEDIR: out   bit;
        DIR:    out     bit;
        CRSE:   out     bit;
        EXDISN: out     bit;
        PRCTR:  out     bit;
        HOLD:   out     bit);

end chp396mod;

architecture chp396 of chp396mod is

 component chp396

   generic (TEMPERATURE: real;  VDD: real);

   port (CLOCK   : in     bit;
      RSTSW   : in     bit;
      SWHOLDN : in     bit;
      EXHOLDN : in     bit;
      FCHFFN  : in     bit;
      LLIMN   : in     bit;
      RLT     : in     bit;
      VHFHI   : in     bit;
      RGT     : in     bit;
      PLT     : in     bit;
      PGT     : in     bit;
      PMAX    : in     bit;
      TC1     : in     bit;
      TUNERST : in     bit;
      NOTUNE  : in     bit;
      SWENAN  : out    bit;
      FINE    : out    bit;
      HOME    : out    bit;
      TUNE    : out    bit;
      RLEGFF  : out    bit;
      NORMAL  : out    bit;
      RSTPUL  : out    bit;
      EXEC    : out    bit;
      ENPCTR  : out    bit;
      LCMD    : out    bit;
      MODSAMPN : out   bit;
      ENPSIG  : out    bit;
      RMINUSN : out    bit;
```

```
      RPLUSN  : out    bit;
      SETDIR  : out    bit;
      PULSEDIR : out   bit;
      DIR     : out    bit;
      CRSE    : out    bit;
      EXDISN  : out    bit;
      PRCTR   : out    bit;
      HOLD    : out    bit);

 end component;

 for NO: chp396 use entity work.chp396(struct);

begin

-------------------------------------------------------------------
-- This section passes through the Voltage VDD and temperature values to the --
-- structure which passes it through generic maps into the gates. This    --
-- method allows the user to change the values as needed without having to  --
-- change anything else in other source code programs.                --
-------------------------------------------------------------------

 NO: chp396
  generic map (TEMPERATURE=>298.15, VDD=> 10.0) -- 298.15 K, 10 Volts

  port map (CLOCK,RSTSW,SWHOLDN,EXHOLDN,FCHFFN,LLIMN,RLT,VHFHI,RGT,PLT,
        PGT,PMAX,TC1,TUNERST,NOTUNE,SWENAN,FINE,HOME,TUNE,RLEGFF,
        NORMAL,RSTPUL,EXEC,ENPCTR,LCMD,MODSAMPN,ENPSIG,RMINUSN,
        RPLUSN,SETDIR,PULSEDIR,DIR,CRSE,EXDISN,PRCTR,HOLD);

end chp396;


use STD.Standard.all;
use work.tables.all;
use work.gem_constants.all;
use work.gem_delays.all;

-------------------------------------------------------------------
-- Test bench for simulation of the SM-B-746396 integrated circuit of the  --
-- PRC-70 radio. The clock cycle is scaled to 4000 ns.                --
-------------------------------------------------------------------

entity test_chp396 is
end test_chp396;

architecture test396 of test_chp396 is

 signal CLOCK      : bit:='0';
 signal RSTSW      : bit:='0';
 signal SWHOLDN    : bit:='1';
 signal EXHOLDN    : bit:='1';
 signal FCHFFN     : bit:='1';
 signal LLIMN      : bit:='1';
```

```
signal RLT          : bit:='0';
signal VHFHI        : bit:='1';
signal RGT          : bit:='0';
signal PLT          : bit:='0';
signal PGT          : bit:='0';
signal PMAX         : bit:='0';
signal TC1          : bit:='1';
signal TUNERST      : bit:='1';
signal NOTUNE       : bit:='1';
signal SWENAN       : bit;
signal FINE         : bit;
signal HOME         : bit;
signal TUNE         : bit;
signal RLEGFF       : bit;
signal NORMAL       : bit;
signal RSTPUL       : bit;
signal EXEC         : bit;
signal ENPCTR       : bit;
signal LCMD         : bit;
signal MODSAMPN     : bit;
signal ENPSIG       : bit;
signal RMINUSN      : bit;
signal RPLUSN       : bit;
signal SETDIR       : bit;
signal PULSEDIR     : bit;
signal DIR          : bit;
signal CRSE         : bit;
signal EXDISN       : bit;
signal PRCTR        : bit;
signal HOLD         : bit;
signal L1           : bit:='0';
signal L2           : bit:='0';
signal CLOCKA       : bit:='0';
signal CLOCKB       : bit:='0';
signal CLOCKC       : bit:='1';
signal STOP         : bit:='0';

component chp396mod
 port (CLOCK  : in    bit;
       RSTSW  : in    bit;
       SWHOLDN : in   bit;
       EXHOLDN : in   bit;
       FCHFFN : in    bit;
       LLIMN  : in    bit;
       RLT    : in    bit;
       VHFHI  : in    bit;
       RGT    : in    bit;
       PLT    : in    bit;
       PGT    : in    bit;
       PMAX   : in    bit;
       TC1    : in    bit;
       TUNERST : in   bit;
       NOTUNE : in    bit;
       SWENAN : out   bit;
```

```
    FINE   : out    bit;
    HOME   : out    bit;
    TUNE   : out    bit;
    RLEGFF : out    bit;
    NORMAL : out    bit;
    RSTPUL : out    bit;
    EXEC   : out    bit;
    ENPCTR : out    bit;
    LCMD   : out    bit;
    MODSAMPN: out   bit;
    ENPSIG : out    bit;
    RMINUSN : out   bit;
    RPLUSN : out    bit;
    SETDIR : out    bit;
    PULSEDIR: out   bit;
    DIR    : out    bit;
    CRSE   : out    bit;
    EXDISN : out    bit;
    PRCTR  : out    bit;
    HOLD   : out    bit);
end component;

for CH: chp396mod use entity work.chp396mod(chp396);

begin  -- input stimuli

RGT <=  '1' after 454000 ns, '0' after 460000 ns, '1' after 500000 ns,
        '0' after 518000 ns, '1' after 536000 ns, '0' after 538000 ns,
        '1' after 862000 ns, '0' after 872000 ns, '1' after 1192000 ns,
        '0' after 1264000 ns, '1' after 1372000 ns, '0' after 1408000 ns,
        '1' after 1452000 ns, '0' after 1460000 ns, '1' after 1522000 ns,
        '0' after 1778000 ns, '1' after 1858000 ns, '0' after 1906000 ns;

RLT <=  '1' after 378000 ns, '0' after 382000 ns, '1' after 622000 ns,
        '0' after 830000 ns, '1' after 1184000 ns, '0' after 1228000 ns,
        '1' after 1336000 ns, '0' after 1370000 ns, '1' after 1410000 ns,
        '0' after 1450000 ns, '1' after 1458000 ns, '0' after 1524000 ns;

VHFHI <= '0' after 1140000 ns, '1' after 1170000 ns, '0' after 1814000 ns,
        '1' after 1904000 ns, '0' after 1906000 ns;

PGT <=  '1' after 1174000 ns, '0' after 1196000 ns, '1' after 1266000 ns,
        '0' after 1300000 ns, '1' after 1414000 ns, '0' after 1462000 ns,
        '1' after 1526000 ns, '0' after 1780000 ns, '1' after 1904000 ns,
        '0' after 1918000 ns, '1' after 1956000 ns;

PLT <=  '1' after 490000 ns, '0' after 566000 ns, '1' after 578000 ns,
        '0' after 608000 ns, '1' after 1136000 ns, '0' after 1162000 ns,
        '1' after 1302000 ns, '0' after 1412000 ns, '1' after 1782000 ns,
        '0' after 1958000 ns;

TUNERST <='0' after 2000 ns, '1' after 574000 ns, '0' after 576000 ns,
        '1' after 618000 ns, '0' after 620000 ns, '1' after 1226000 ns,
        '0' after 1228000 ns, '1' after 1920000 ns, '0' after 1922000 ns;
```

```
PCHFFN <='0' after 2000 ns, '1' after 4000 ns, '0' after 198000 ns,
      '1' after 200000 ns;

TC1 <=  '0' after 352000 ns, '1' after 388000 ns, '0' after 834000 ns,
      '1' after 848000 ns, '0' after 1108000 ns, '1' after 1136000 ns,
      '0' after 1728000 ns;

PMAX <= '1' after 284000 ns, '0' after 364000 ns, '1' after 844000 ns,
      '0' after 902000 ns, '1' after 1506000 ns, '0' after 1528000 ns;

RSTSW <= '1' after 250000 ns, '0' after 364000 ns, '1' after 428000 ns,
      '0' after 628000 ns, '1' after 900000 ns;

EXHOLDN <='0' after 230000 ns, '1' after 240000 ns;

SWHOLDN <='0' after 238000 ns, '1' after 246000 ns;

NOTUNE <='0' after 2000 ns, '1' after 196000 ns, '0' after 198000 ns,
      '1' after 830000 ns, '0' after 832000 ns;

LLIMN <= '0' after 418000 ns, '1' after 444000 ns, '0' after 576000 ns,
      '1' after 604000 ns, '0' after 910000 ns, '1' after 1106000 ns,
      '0' after 1538000 ns, '1' after 1726000 ns, '0' after 1760000 ns,
      '1' after 1772000 ns;

L1 <=  '1' after 4000 ns, '0' after 194000 ns,
      '1' after 288000 ns, '0' after 350000 ns,
      '1' after 384000 ns, '0' after 420000 ns,
      '1' after 432000 ns, '0' after 442000 ns,
      '1' after 492000 ns, '0' after 498000 ns,
      '1' after 520000 ns, '0' after 534000 ns,
      '1' after 580000 ns, '0' after 606000 ns,
      '1' after 836000 ns, '0' after 842000 ns,
      '1' after 852000 ns, '0' after 858000 ns,
      '1' after 864000 ns, '0' after 870000 ns,
      '1' after 912000 ns, '0' after 1134000 ns,
      '1' after 1144000 ns, '0' after 1166000 ns,
      '1' after 1176000 ns, '0' after 1190000 ns,
      '1' after 1268000 ns, '0' after 1298000 ns,
      '1' after 1304000 ns, '0' after 1334000 ns,
      '1' after 1416000 ns, '0' after 1446000 ns,
      '1' after 1464000 ns, '0' after 1502000 ns,
      '1' after 1540000 ns, '0' after 1758000 ns,
      '1' after 1784000 ns, '0' after 1854000 ns,
      '1' after 1924000 ns, '0' after 1954000 ns,
      '1' after 1960000 ns, '0' after 1988000 ns;

L2 <=  '1' after 202000 ns, '0' after 228000 ns,
      '1' after 246000 ns, '0' after 280000 ns,
      '1' after 354000 ns, '0' after 376000 ns,
      '1' after 446000 ns, '0' after 452000 ns,
      '1' after 462000 ns, '0' after 488000 ns,
      '1' after 502000 ns, '0' after 516000 ns,
```

```
        '1' after 538000 ns, '0' after 564000 ns,
        '1' after 610000 ns, '0' after 616000 ns,
        '1' after 630000 ns, '0' after 828000 ns,
        '1' after 874000 ns, '0' after 896000 ns,
        '1' after 1198000 ns, '0' after 1224000 ns,
        '1' after 1230000 ns, '0' after 1260000 ns,
        '1' after 1338000 ns, '0' after 1368000 ns,
        '1' after 1374000 ns, '0' after 1404000 ns,
        '1' after 1510000 ns, '0' after 1520000 ns,
        '1' after 1530000 ns, '0' after 1536000 ns,
        '1' after 1762000 ns, '0' after 1768000 ns,
        '1' after 1862000 ns, '0' after 1902000 ns,
        '1' after 1910000 ns, '0' after 1916000 ns;

CLOCKA <='1' after 194000 ns, '0' after 202000 ns,
        '1' after 228000 ns, '0' after 232000 ns,
        '1' after 234000 ns, '0' after 236000 ns,
        '1' after 242000 ns, '0' after 244000 ns,
        '1' after 280000 ns, '0' after 282000 ns,
        '1' after 286000 ns, '0' after 288000 ns,
        '1' after 350000 ns, '0' after 354000 ns,
        '1' after 376000 ns, '0' after 384000 ns,
        '1' after 418000 ns, '0' after 420000 ns,
        '1' after 424000 ns, '0' after 426000 ns,
        '1' after 430000 ns, '0' after 432000 ns,
        '1' after 442000 ns, '0' after 446000 ns,
        '1' after 452000 ns, '0' after 456000 ns,
        '1' after 458000 ns, '0' after 462000 ns,
        '1' after 488000 ns, '0' after 492000 ns,
        '1' after 498000 ns, '0' after 502000 ns,
        '1' after 516000 ns, '0' after 520000 ns,
        '1' after 534000 ns, '0' after 538000 ns,
        '1' after 564000 ns, '0' after 570000 ns,
        '1' after 572000 ns, '0' after 580000 ns,
        '1' after 606000 ns, '0' after 610000 ns,
        '1' after 616000 ns, '0' after 624000 ns,
        '1' after 626000 ns, '0' after 630000 ns,
        '1' after 828000 ns, '0' after 836000 ns,
        '1' after 842000 ns, '0' after 846000 ns,
        '1' after 850000 ns, '0' after 852000 ns,
        '1' after 858000 ns, '0' after 864000 ns,
        '1' after 870000 ns, '0' after 874000 ns,
        '1' after 896000 ns, '0' after 898000 ns,
        '1' after 904000 ns, '0' after 906000 ns,
        '1' after 908000 ns, '0' after 912000 ns,
        '1' after 1134000 ns, '0' after 1138000 ns,
        '1' after 1142000 ns, '0' after 1144000 ns,
        '1' after 1166000 ns, '0' after 1176000 ns,
        '1' after 1190000 ns, '0' after 1198000 ns,
        '1' after 1224000 ns, '0' after 1230000 ns,
        '1' after 1260000 ns, '0' after 1268000 ns,
        '1' after 1298000 ns, '0' after 1304000 ns,
        '1' after 1334000 ns, '0' after 1338000 ns,
        '1' after 1368000 ns, '0' after 1374000 ns,
```

```
        '1' after 1404000 ns, '0' after 1416000 ns,
        '1' after 1446000 ns, '0' after 1454000 ns,
        '1' after 1456000 ns, '0' after 1464000 ns,
        '1' after 1502000 ns, '0' after 1504000 ns,
        '1' after 1508000 ns, '0' after 1510000 ns,
        '1' after 1520000 ns, '0' after 1530000 ns,
        '1' after 1536000 ns, '0' after 1540000 ns,
        '1' after 1758000 ns, '0' after 1762000 ns,
        '1' after 1768000 ns, '0' after 1770000 ns,
        '1' after 1774000 ns, '0' after 1784000 ns,
        '1' after 1854000 ns, '0' after 1856000 ns,
        '1' after 1860000 ns, '0' after 1862000 ns,
        '1' after 1900000 ns, '0' after 1902000 ns,
        '1' after 1908000 ns, '0' after 1910000 ns,
        '1' after 1916000 ns, '0' after 1924000 ns,
        '1' after 1954000 ns, '0' after 1960000 ns;

  CLOCKB <= not CLOCKB after 2000 ns;

  CLOCKC <= not CLOCKC after 2000 ns;

 STOP <= '1' after 2000000 ns; -- stop time when simulation ends.

 CH: chp396mod
  port map (CLOCK,RSTSW,SWHOLDN,EXHOLDN,PCHFFN,LLIMN,RLT,VHFHI,RGT,PLT,
        PGT,PMAX,TC1,TUNERST,NOTUNE,SWENAN,FINE,HOME,TUNE,RLEGFF,
        NORMAL,RSTPUL,EXEC,ENPCTR,LCMD,MODSAMPN,ENPSIG,RMINUSN,
        RPLUSN,SETDIR,PULSEDIR,DIR,CRSE,EXDISN,PRCTR,HOLD);

P1: process -- process to incorporate a perpetual running clock into the test
   begin

    wait for 1 fs;
    if L1 = '1' then
CLOCK <= CLOCKB;
    elsif L2 = '1' then
     CLOCK <= CLOCKC;
    else
     CLOCK <= CLOCKA;
    end if;

    wait on L1, L2, CLOCKA, CLOCKB, CLOCKC;

   end process P1;

P2: process -- process to stop the simulation at a specified time.
   begin
    wait until STOP = '1';
    assert STOP = '0'
      report "end of simulation after 2 ms"
      severity FAILURE;
   end process P2;

end test396;
```

ARMY RESEARCH LABORATORY
ELECTRONICS AND POWER SOURCES DIRECTORATE
CONTRACT OR IN-HOUSE TECHNICAL REPORTS
MANDATORY DISTRIBUTION LIST

Defense Technical Information Center*
ATTN: DTIC-OCC
Cameron Station (Bldg 5)
Alexandria, VA 22304-6145
(*Note: Two copies will be sent from STINFO office, Fort Monmouth, NJ)

Director
US Army Material Systems Analysis Actv
ATTN: DRXSY-MP
(1) Aberdeen Proving Ground, MD 21005

Commander, AMC
ATTN: AMCDE-SC
5001 Eisenhower Ave.
(1) Alexandria, VA 22333-0001

Director
Army Research Laboratory
ATTN: AMSRL-D (John W. Lyons)
2800 Powder Mill Road
(1) Adelphi, MD 20783-1145

Director
Army Research Laboratory
ATTN: AMSRL-DD (COL Thomas A. Dunn)
2800 Powder Mill Road
(1) Adelphi, MD 20783-1145

Director
Army Research Laboratory
2800 Powder Mill Road
Adelphi, MD 20783-1145
(1) AMSRL-OP-CI-AD (Tech Pubs)
(1) AMSRL-OP-CI-AD (Records Mgt)
(1) AMSRL-OP-CI-AD (Tech Library)

Directorate Executive
Army Research Laboratory
Electronics and Power Sources Directorate
Fort Monmouth, NJ 07703-5601
(1) AMSRL-EP
(1) AMSRL-EP-T (M. Howard)
(1) AMSRL-OP-RM-FM
(22) Originating Office

Advisory Group on Electron Devices
ATTN: Documents
2011 Crystal Drive, Suite 307
(2) Arlington, VA 22202

Commander, CECOM
R&D Technical Library
Fort Monmouth, NJ 07703-5703
(1) AMSEL-IM-BM-I-L-R (Tech Library)
(3) AMSEL-IM-BM-I-L-R (STINFO ofc)

Deputy for Science & Technology
Office, Asst Sec Army (R&D)
(1) Washington, DC 20310

HQDA (DAMA-ARZ-D/
Dr. F.D. Verderame)
(1) Washington, DC 20310

Director
Naval Research Laboratory
ATTN: Code 2627
(1) Washington, DC 20375-5000

USAF Rome Laboratory Technical Library
ATTN: Documents Library
FL2810, Corridor W, Ste 262, RL/SUL
26 Electronics Parkway, Bldg 106
(1) Griffiss Air Force Base, NY 13441-4514

Dir, ARL Battlefield
Environment Directorate
ATTN: AMSRL-BE
White Sands Missile Range
(1) NM 88002-5501

Dir, ARL Sensors, Signatures,
Signal & Information Processing
Directorate (S3I)
ATTN: AMSRL-SS
2800 Powder Mill Road
(1) Adelphi, MD 20783-1145

Dir, CECOM Night Vision/
Electronic Sensors Directorate
ATTN: AMSEL-RD-NV-D
(1) Fort Belvoir, VA 22060-5677

Dir, CECOM Intelligence and
Electronic Warfare Directorate
ATTN: AMSEL-RD-IEW-D
Vint Hill Farms Station
(1) Warrenton, VA 22186-5100

Cdr, Marine Corps Liaison Office
ATTN: AMSEL-LN-MC
(1) Fort Monmouth, NJ 07703-5703